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) [5] 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 [11] and [6]. Sanadhya and Sarkar [16] 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. [11], who mentioned an 18-step collision for SHA-256 which was recently corrected in [12]. The work [11] also provided a differential path for 19-step near collision for SHA-256. An earlier work [10] studied a very simplified variant of

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

S.M. Bellovin et al. (Eds.): ACNS 2008, LNCS 5037, pp. 130–143, 2008.

SHA-256. The encryption mode of SHA-256 is analyzed in [23] and is not relevant to collision search attacks. Recently, at FSE '08, Nikolić and Biryukov [13] 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 [16]. 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 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: $\{m_0, m_1, m_2, \ldots m_{15}\}$ and $\{m'_0, m'_1, m'_2, \ldots m'_{15}\}$.
- The expanded message pair is: $\{W_0, W_1, W_2, \ldots W_{63}\}$ and $\{W'_0, W'_1, W'_2, \ldots W'_{63}\}$.
- $\oplus$: bitwise XOR.
- $+$: addition modulo 2^{32}.
- $\Delta W_i = W_i \oplus W'_i$
- $\mathrm{ROTR}^n(x)$: Right rotation of a 32 bit quantity x by n bits.
- $\mathrm{SHR}^n(x)$: 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 [18]. 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 [18].

The round function of SHA-256 hash function uses 8 registers. 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 "`Choice`" and "`Majority`" respectively. The functions Σ_0 and Σ_1 are defined as:

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

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, $\ldots m_{15}$, 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 σ_0 and σ_1 are defined as:

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

The IV = (a_{-1}, b_{-1}, c_{-1}, d_{-1}, e_{-1}, f_{-1}, g_{-1}, h_{-1}) is defined by 8 32-bit constants. 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. 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 $\{\Delta W_0, \Delta W_1, \ldots \Delta W_{15}\}$ is likely to generate colliding pairs with probability $\frac{1}{2^8}$ then one needs to try roughly 2^8 different pairs $\{W_0, W_1, \ldots, W_{15}\}$ and $\{W'_0, W'_1, \ldots W'_{15}\}$ 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 [22,20] and neutral bit technique [1] are the two widely used methods to find colliding message pairs.

For a fuller discussion of linearized local collisions and differential paths, refer to [17]. We next discuss the SHA-256 linearized local collisions.

4.1 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 [5] and the second is one of the 16 local collisions presented in [16]. From among the 16, we choose the 5^{th} 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^{th}, 14^{th} and 16^{th}).
2. It has the highest probability among these 4.

We call the two local collisions the GH local collision and the SS_5 local collision respectively. The other three local collisions from [16] are denoted by SS_7, SS_{14} and SS_{16}.

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 [11].
3. In SS_5, 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 [16] 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_5 : $\{x, \Sigma_0(x) \oplus \Sigma_1(x),\ \Sigma_0(\Sigma_1(x)),\ \Sigma_0(x) \oplus \Sigma_1(x),\ 0,\ \Sigma_0(x) \oplus \Sigma_1(x),\ 0,\ 0,\ x\}$
- 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, Σ_0 and Σ_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.

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 [11] and mentioned in [16]. 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 $\Delta W_0 = \Delta W_1 = \ldots = \Delta W_{i-1} = \Delta W_{i+9} = \Delta W_{i+10} = \ldots = \Delta W_{15} = 0$, 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 :

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

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

Let a local collision $\mathcal{L}$ start at Step 3 and hence end at Step 11. This local collision defines the 9 word differences $\Delta W_3, \Delta W_4, \ldots \Delta W_{11}$. The first step of the local collision corresponds to ΔW_3 and the 9^{th} step corresponds to ΔW_{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.

ΔW_9 corresponds to the 7^{th} step word difference for $\mathcal{L}$. If $\mathcal{L}$ is chosen such that it's 7^{th} 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.

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

Step i	ΔW_i	Δa_i	Δb_i	Δc_i	Δd_i	Δe_i	Δf_i	Δg_i	Δh_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

Similarly, if the 8^{th} 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^{th} and the 8^{th} 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.

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 [7], Table 3. We next highlight the advantages of our conditions with those in [7].

1. The conditions in [7] are for only 9-step collision in SHA-256. Our conditions are for 18-step collision in SHA-256.

Table 2. Conditions for the 18 step differential path in Table 1. x^i denotes i^{th} bit of a 32 bit quantity x. $\overline{x}$ 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	$b_3^{31} = c_3^{31}$	$\frac{1}{2}$	$f_3^{31} = g_3^{31}$	$\frac{1}{2}$	$\overline{W_4^i} = (\Sigma_0(a_3))^i;\ i = 9, 18, 29$ $W_4^i = (\Sigma_1(e_3))^i;\ i = 6, 20, 25$	$\frac{1}{2^6}$	bit differences $\Delta W_4^i; i = 9, 18, 29$ propagate into e_4	$\frac{1}{2^3}$	$\frac{1}{2^{11}}$
5	$a_4^{31} = \overline{c_4^{31}}$	$\frac{1}{2}$	$e_4^{31} = 1,$ $e_3^i = f_3^i;$ $i = 9, 18, 29$	$\frac{1}{2^4}$	$\overline{W_5^i} = (\Sigma_1(e_4))^i;$ $i = 3, 4, 7, 12, 16,$ $18, 23, 25, 30$	$\frac{1}{2^9}$	-	-	$\frac{1}{2^{14}}$
6	$a_5^{31} = b_5^{31}$	$\frac{1}{2}$	$e_5^{31} = 1;$ $i = 9, 18, 29$ $e_4^{31} = \overline{e_5^{31}} * e_3^{31}$	$\frac{1}{2^4}$	$\overline{W_5^i} = (\Sigma_1(e_5) + f_5)^i;$ $i = 6, 9, 18, 20, 25, 29$	$\frac{1}{2^6}$	-	-	$\frac{1}{2^{11}}$
7	-	-	$e_6^i = 1;$ $i = 9, 18, 29, 31$	$\frac{1}{2^4}$	-	-	-	-	$\frac{1}{2^4}$
8	-	-	$e_6^{31} = e_7^{31} * e_5^{31}$	$\frac{1}{2}$	$\overline{W_8^i} = (\Sigma_1(e_7) + h_7)^i;$ $i = 6, 9, 18, 20, 25, 29$	$\frac{1}{2^6}$	-	-	$\frac{1}{2^7}$
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.	$\frac{1}{2^{49}}$

2. The GH local collision is used in [7] whereas we use SS_5 local collision. Further, no explanation is provided in [7] 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 [7] 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.

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 Δa_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 Σ_0 (resp. Σ_1) expands this difference to 3 bit positions 6, 20 and 25 (resp. 9, 18 and 29). The word difference ΔW_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 [21]. We use XOR differences only, unlike [21] 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 Σ_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 Σ_0 and Σ_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:

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

In these two computations, bits 6, 20 and 25 corresponding to Σ_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 Δe_4. Since $d_3 = d'_3$, $h_3 = h'_3$ and $f_{IF}(e_3, f_3, g_3) = f_{IF}(e'_3, f'_3, g'_3)$;

$$\text{if we write} \quad rest = \Sigma_1(e_3) + f_{IF}(e_3, f_3, g_3) + h_4 + K_4,$$

$$\begin{aligned} &\text{then} && e_4 = rest + W_4, \\ &\text{and} && e'_4 = rest + W'_4. \end{aligned}$$

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 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 $W_4^{10} = W_4^{11} = \ldots = W_4^{17} = 0$.
4. if $rest^{18}$ is 1 then $W_4^{10} = W_4^{11} = \ldots = W_4^{17} = 1$.
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. An example of such a pseudo collision in provided in [17].

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_1	0-7	ccea5c17	53ad1a2d	141db23c	b6acfaa8	5ee7fe4d	53c5b764	2bf20d44	87d63bf6
	8-15	63a07869	f305fdea	26ee271f	b973b91c	d0f87828	b724a487	a295fa2a	0a67c97a
M_2	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_1	0-7	ed919421	aa75e4fe	8548d0e0	9c1888f7	1da3fc3d	a11f7a02	bb463b64	e9b28365
	8-15	323ecf28	8097e497	4343b78b	dc484e91	bf588b4b	8401140a	42499da1	f88a3e2e
M_2	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 [17].

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

In [15] and [14] coding theoretic techniques were used to search for differential paths in SHA-1. Extension to SHA-2 was mentioned in [11]. 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 ΔW_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 [3] type disturbance vector approach. Let DV = $\{d_0, d_1, d_2, \ldots, d_{255}\}$ be a 256-bit disturbance vector. If $d_i = 1$ then the two initial messages differ in their i^{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 ΔW_i $(i \geq 16)$ in terms of $d_0, \ldots, d_{255}$. 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 $(d_0, \ldots, d_{255})$ 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 ΔW_i $(i \geq 16)$ in terms of $d_0, \ldots, d_{255}$. 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 $(d_0, 0, \ldots, 0)$, 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 $i \geq 16$, ΔW_i is now obtained using the message recursion. This expresses all the ΔW_is $(i \geq 16)$ as linear function of d_0. Next consider the 32-bit vector $(0, d_1, 0, \ldots, 0)$; use $\mathcal{L}$ to obtain the next eight words and the message expansion recursion to express ΔW_is $(i \geq 16)$ as linear function of d_1. Now, for the 32-bit vector $(d_0, d_1, 0, \ldots, 0)$, we can express ΔW_is $(i \geq 16)$ 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 $(d_0, \ldots, d_{255})$. The exact details are given in Table 5.

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

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

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

```
For(i = 0 to 8){
    For(j = 0 to 31){
        set D := (0, 0, ..., d_{32i+j}, 0, ..., 0);        /*  The j^th bit of D is given by d_{32i+j}.
                  Each d_n ∈ {0,1} is the component of the disturbance vector and D ∈ {0,1}^32  */
        set δW_cur := LC(D);
        For(k = i to i + 8){
            ΔW_final[k] = ΔW_final[k] ⊕ δW_cur[k − i];
        }
    }
}
/* At this point the ΔW_final list contains W_i ⊕ W'_i for 0 ≤ i < 16 */
```

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

Methods presented in [2], [8] and [19] 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.

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 [2], [8] and [19]. 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 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 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 [11]. As has already been noted in [11], the GH local collision causes certain impossible conditions in the search for actual colliding pairs. The use of 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 32×256, 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 [17].

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 [2], [8] and [19] are used.

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

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 [9].

References

1. Biham, E., Chen, R.: Near-Collisions of SHA-0. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 290–305. Springer, Heidelberg (2004)
2. Canteaut, A., Chabaud, F.: 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)
3. Chabaud, F., Joux, A.: Differential Collisions in SHA-0. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 56–71. Springer, Heidelberg (1998)
4. Cramer, R.J.F. (ed.): EUROCRYPT 2005. LNCS, vol. 3494. Springer, Heidelberg (2005)
5. Gilbert, H., Handschuh, H.: Security Analysis of SHA-256 and Sisters. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 175–193. Springer, Heidelberg (2004)
6. P. Hawkes, M. Paddon, G.G. Rose. On Corrective Patterns for the SHA-2 Family. Cryptology eprint Archive (August 2004), `http://eprint.iacr.org/2004/207`
7. Hölbl, M., Rechberger, C., Welzer, T.: Finding Message Pairs Conforming to Simple SHA-256 Characteristics. In: Lucks, S., Sadeghi, A.-R., Wolf, C. (eds.) Stefan Lucks, Ahmad-Reza Sadeghi, and Christopher Wolf, editors, Preliminary Proceeding Records of WEWoRC 2007 - Western European Workshop on Research in Cryptology, Bochum, Germany, pp. 21–25 (2007), `http://www.hgi.rub.de/weworc07/PreliminaryConferenceRecord.pdf`
8. Leon, J.S.: A Probabilistic Algorithm for Computing Minimum Weights of Large Error-Correcting Codes. IEEE Transactions on Information Theory 34(5), 1354–1359 (1988)
9. Matusiewicz, K.: Analysis of Modern Dedicated Cryptographic Hash Functions. PhD thesis, Macquarie University (August 2007), `http://www.ics.mq.edu.au/~kmatus/Matusiewicz-PhDthesis.pdf`

10. Matusiewicz, K., Pieprzyk, J., Pramstaller, N., Rechberger, C., Rijmen, V.: Analysis of simplified variants of SHA-256. In: Wolf, C., Lucks, S., Yau, P.-W. (eds.) WEWoRC 2005 - Western European Workshop on Research in Cryptology, Leuven, Belgium, July 5-7, 2005. LNI, vol. 74, pp. 123–134. GI (2005)
11. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: Analysis of Step-Reduced SHA-256. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 126–143. Springer, Heidelberg (2006)
12. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: Analysis of Step-Reduced SHA-256. Cryptology eprint Archive (March 2008), http://eprint.iacr.org/2008/130
13. Nikolić, I., Biryukov, A.: Collisions for Step-Reduced SHA-256. In: Nyberg, K. (ed.) Fast Software Encryption 2008. volume Pre-proceedings version of Lecture Notes in Computer Science, pp. 1–16. Springer, Heidelberg (2008)
14. Pramstaller, N., Rechberger, C., Rijmen, V.: Exploiting Coding Theory for Collision Attacks on SHA-1. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 78–95. Springer, Heidelberg (2005)
15. Rijmen, V., Oswald, E.: Update on SHA-1. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 58–71. Springer, Heidelberg (2005)
16. Sanadhya, S.K., Sarkar, P.: New Local Collisions for the SHA-2 Hash Family. In: Nam, K.-H., Rhee, G. (eds.) ICISC 2007. LNCS, vol. 4817, pp. 193–205. Springer, Heidelberg (2007)
17. Sanadhya, S.K., Sarkar, P.: Attacking Reduced Round SHA-256. Cryptology eprint Archive (March 2008), http://eprint.iacr.org/2008/142
18. Secure Hash Standard. Federal Information Processing Standard Publication 180-2. U.S. Department of Commerce, National Institute of Standards and Technology (NIST) (2002), http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf
19. Stern, J.: A Method for Finding Codewords of Small Weight. In: Wolfmann, J., Cohen, G. (eds.) Coding Theory 1988. LNCS, vol. 388, pp. 106–113. Springer, Heidelberg (1989)
20. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: Cramer [4], pp. 1–18
21. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
22. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer [4], pp. 19–35
23. Yoshida, H., Biryukov, A.: Analysis of a SHA-256 Variant. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 245–260. Springer, Heidelberg (2006)