New Collision attacks Against Up To 24-step SHA-2

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. In this work, we provide new and improved attacks against 22, 23 and 24-step SHA-2 family using a local collision given by Sanadhya and Sarkar (SS) at ACISP '08. The success probability of our 22-step attack is 1 for both SHA-256 and SHA-512. The computational efforts for the 23-step and 24-step SHA-256 attacks are respectively $2^{11.5}$ and $2^{28.5}$ calls to the corresponding step reduced SHA-256. The corresponding values for the 23 and 24-step SHA-512 attack are respectively $2^{16.5}$ and $2^{32.5}$ calls. Using a look-up table having 2^{32} (resp. 2^{64}) entries the computational effort for finding 24-step SHA-256 (resp. SHA-512) collisions can be reduced to $2^{15.5}$ (resp. $2^{22.5}$) calls. We exhibit colliding message pairs for 22, 23 and 24-step SHA-256 and SHA-512. This is the *first* time that a colliding message pair for 24-step SHA-512 is provided. The previous work on 23 and 24-step SHA-2 attacks is due to Indesteege et al. and utilizes the local collision presented by Nikolić and Biryukov (NB) at FSE '08. The reported computational efforts are 2^{18} and $2^{28.5}$ for 23 and 24-step SHA-256 respectively and $2^{43.9}$ and 2^{53} for 23 and 24-step SHA-512. The previous 23 and 24-step attacks first constructed a pseudo-collision and later converted it into a collision for the reduced round SHA-2 family. We show that this two step procedure is unnecessary. Although these attacks improve upon the existing reduced round SHA-2 attacks, they do not threaten the security of the full SHA-2 family.

Keywords: Cryptanalysis, SHA-2 hash family, reduced round attacks

1 Introduction

Cryptanalysis of SHA-2 family has recently gained momentum due to the important work of Nikolić and Biryukov [5]. Prior work on finding collisions for step reduced SHA-256 was done in [3, 4] and [7]. These earlier works used local collisions valid for the XOR linearized version of SHA-256 from [1] and [6]. On the other hand, the work [5] used a local collision which is valid for the actual SHA-256.

The authors in [5] developed techniques to handle nonlinear functions and the message expansion of SHA-2 to obtain collisions for up to 21-step SHA-256. The 21-step attack of [5] succeeded with probability 2^{-19}. Using similar techniques, but utilizing a different local collision, [9] showed an attack against 20-step SHA-2 which succeeds with probability one and an attack against 21-step SHA-256 which succeeds with probability 2^{-15}. Further work [8] developed collision attacks against 21-step SHA-2 family which succeeds with probability one. Very recently, Indesteege et al. [2] have developed attacks against 23 and 24 step SHA-2 family. They utilize the local collision from [5] in these attacks.

Our Contributions. Our contributions in terms of the number of steps attacked and the success probability of these attacks are as follows.

- We describe the first *deterministic* attack against 22-step SHA-256 and SHA-512.
- We describe new attacks against 23 and 24-step SHA-256 and SHA-512.
  - The complexity of the 23-step attack for both SHA-256 and SHA-512 is improved in comparison to the existing 23-step attacks of [2].

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

- The complexity of 24-step SHA-512 attack is improved in comparison to the existing attack of [2]. In fact, improving the complexity to $2^{32.5}$ from the earlier reported 2^{53} allows us to provide the *first* message pair which collides for 24-step SHA-512.
- Using a table lookup, the complexity of the 24-step SHA-256 attack is improved in comparison to the existing 24-step attack of [2]. The table contains 2^{32} entries with each entry of size 8 bytes. Similary, the complexity of the 24-step SHA-512 attack is also improved using a table lookup. For this case, the table lookup has 2^{64} entries each entry of 16 bytes.
- Examples of Colliding message pairs are provided for 22, 23 and 24-step SHA-256 and SHA-512.

Our contributions to the methodology of the attacks are as follows.

– We use a different local collision for our 22, 23 and 24-step attacks. The earlier work [2] uses the local collision from [5] while we use a local collision from [9].
– The work in [2] describes 23 and 24-step collisions as a two-part procedure– first obtain a pseudo-collision and then convert it into a collision. In contrast, our analysis is direct and shows that such a two-part description is unnecessary.
– Details of a required "guess-then-determine algorithm" to solve a non-linear equation arising in the 24-step attack are provided in this work. A suggestion for a similar algorithm is given in [2] but no details are provided. There are two algorithms– one for SHA-256 and the other for SHA-512.

A summary of results on collision attacks against reduced SHA-2 family is given in Table 1.

Table 1. Summary of results against reduced SHA-2 family. Effort is expressed as either the probability of success or as the number of calls to the respective reduced round hash function.

Work	Hash Function	Steps	Effort		Local Collision	Attack Type	Example
			Prob.	Calls	utilized		provided
[3, 4]	SHA-256	18		*	GH [1]	Linear	yes
[7]	SHA-256	18		**	SS_5 [6]	"	yes
[5]	SHA-256	20	$\frac{1}{3}$		NB [5]	Non-linear	yes
		21	2^{-19}		"	"	yes
[9]	SHA-256/SHA-512	18,20	1	1	SS [9]	"	yes
	SHA-256	21	2^{-15}		"	"	yes
[8]	SHA-256/SHA-512	21	1	1	"	"	yes
[2]	SHA-256	23		2^{18}	NB [5]	"	yes
		24		$2^{28.5}$	"	"	yes
	SHA-512	23		$2^{43.9}$	"	"	yes
		24		2^{53}	"	"	**no**
This work	SHA-256/SHA-512	22	1	1	SS [9]	"	yes
	SHA-256	23		$2^{11.5}$	"	"	yes
		24		$2^{28.5}$	"	"	yes
		24		$2^{15.5}$ †	"	"	no
	SHA-512	23		$2^{16.5}$	"	"	yes
		24		$2^{32.5}$	"	"	**yes**
		24		$2^{22.5}$ ‡	"	"	no

* It is mentioned in [3, 4] that the effort is 2^0 but no details are provided.
** Effort is given as running a C-program for about 30–40 minutes on a standard PC.
† A table containing 2^{32} entries, each entry of size 8 bytes, is required.
‡ A table containing 2^{64} entries, each entry of size 16 bytes, is required.

2 Preliminaries

In this paper we use the following notation:
- Message words: $W_i \in \{0,1\}^n$, $W'_i \in \{0,1\}^n$; n is 32 for SHA-256 and 64 for SHA-512.
- Colliding message pair: $\{W_0, W_1, W_2, \ldots W_{15}\}$ and $\{W'_0, W'_1, W'_2, \ldots W'_{15}\}$.
- Expanded message pair: $\{W_0, W_1, W_2, \ldots W_{r-1}\}$ and $\{W'_0, W'_1, W'_2, \ldots W'_{r-1}\}$. The number of steps r is 64 for SHA-256 and 80 for SHA-512.
- The internal registers for the two messages at step i: $\text{REG}_i = \{a_i, \ldots, h_i\}$ and $\text{REG}'_i = \{a'_i, \ldots, h'_i\}$.
- $\text{ROTR}^k(x)$: Right rotation of an n-bit string x by k bits.
- $\text{SHR}^k(x)$: Right shift of an n-bit string x by k bits.
- $\oplus$: bitwise XOR; $+, -$: addition and subtraction modulo 2^n.
- $\delta X = X' - X$ where X is an n-bit quantity.
- $\delta\Sigma_1(x) = \Sigma_1(e'_i) - \Sigma_1(e_i) = \Sigma_1(e_i + x) - \Sigma_1(e_i)$.
- $\delta\Sigma_0(x) = \Sigma_0(a'_i) - \Sigma_0(a_i) = \Sigma_0(a_i + x) - \Sigma_0(a_i)$.
- $\delta f^i_{MAJ}(x, y, z) = f_{MAJ}(a_i + x, b_i + y, c_i + z) - f_{MAJ}(a_i, b_i, c_i)$.
- $\delta f^i_{IF}(x, y, z) = f_{IF}(e_i + x, f_i + y, g_i + z) - f_{IF}(e_i, f_i, g_i)$.

2.1 SHA-2 Hash Family

Eight registers are used in the evaluation of SHA-2. 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). For more details, see § A.

By the form of the step update function, we have the following relation.

Cross Dependence Equation (CDE).

$$e_i = a_i + a_{i-4} - \Sigma_0(a_{i-1}) - f_{MAJ}(a_{i-1}, a_{i-2}, a_{i-3}). \tag{1}$$

Later, we make extensive use of this relation. Note that a special case of this equation was first utilized in §6.1 of [9]. The equation in the form above was used in [8]. This equation can be used to show that the SHA-2 state update can be rewritten in terms of only one state variable. This fact was later observed in [2] independently.

3 Nonlinear Local Collision for SHA-2

We use two variations of a 9-step non-linear local collision for our attacks. This local collision was given recently by Sanadhya and Sarkar [9]. This local collision starts by introducing a perturbation message difference of 1 in the first message word. Next eight message words are chosen suitably to obtain the desired differential path. Table 2 shows the local collision used. The message word differences are different for the two variations of the local collision. Columns headed I and II under δW_i in Table 2 show the message word differences for the first and the second variations of the local collision respectively.

In the local collision, the registers (a_{i-1}, ..., h_{i-1}) and W_i are inputs to Step i of the hash evaluation and this step outputs the registers (a_i, ..., h_i).

3.1 Conditions on the Differential Path of Column I of Table 2

The message word differences are computed from the following equations:

$$\delta W_{i+1} = -1 - \delta f^i_{IF}(1, 0, 0) - \delta\Sigma_1(1), \tag{2}$$

$$\delta W_{i+2} = -1 - \delta f^{i+1}_{IF}(-1, 1, 0) - \delta\Sigma_1(-1), \tag{3}$$

Table 2. The 9-step Sanadhya-Sarkar local collision [9] used in the present work. Our deterministic 22-step attack and the probabilistic 23 and 24-step attacks use unequal message word differences to achieve the same differential path.

Step	δW_i		Register differences							
	I	II	δa_i	δb_i	δc_i	δd_i	δe_i	δf_i	δg_i	δh_i
$i-1$	0	0	0	0	0	0	0	0	0	0
i	1	1	1	0	0	0	1	0	0	0
$i+1$	-1	δW_{i+1}	0	1	0	0	-1	1	0	0
$i+2$	δW_{i+2}	0	0	0	1	0	-1	-1	1	0
$i+3$	δW_{i+3}	δW_{i+3}	0	0	0	1	0	-1	-1	1
$i+4$	0	0	0	0	0	0	1	0	-1	-1
$i+5$	0	0	0	0	0	0	0	1	0	-1
$i+6$	0	0	0	0	0	0	0	0	1	0
$i+7$	δW_{i+7}	0	0	0	0	0	0	0	0	1
$i+8$	-1	-1	0	0	0	0	0	0	0	0

$$\delta W_{i+3} = -\delta f_{IF}^{i+2}(-1,-1,1) - \delta\Sigma_1(-1), \tag{4}$$

$$\delta W_{i+7} = -\delta f_{IF}^{i+6}(0,0,1), \tag{5}$$

$$\delta W_{i+4} = \delta W_{i+5} = \delta W_{i+6} = 0. \tag{6}$$

Further, we will require that the value of δW_{i+1} is -1 and the value of δW_{i+7} is either 0 or 1.

Intermediate registers need to satisfy conditions given in Table 3. These conditions can be derived in the same way as in [5]. The same differential path has also been used by Sanadhya and Sarkar recently to attack 20-step SHA-256/512 in [9] and 21-step SHA-256/512 in [8], both the attacks holding with probability one.

Table 3. Values of a and e register for the δWs given by Column (I) of Table 2 to hold. We have $\beta = \overline{\alpha}$ and using CDE, $\lambda = \beta + \alpha - \Sigma_0(\beta) - f_{MAJ}(\beta, -1, \alpha) = -\Sigma_0(\overline{\alpha})$. The value of u is either 0 or 1. Thus, the independent quantities are α, γ and μ.

index	$i-2$	$i-1$	i	$i+1$	$i+2$	$i+3$	$i+4$	$i+5$	$i+6$
a	α	α	-1	β	β				
e	γ	$\gamma+1$	-1	μ	λ	$\lambda-1$	-1	-1	$-1-u$

The values shown in Table 3 have been chosen so that the conditions on δW_{i+1} and δW_{i+5} to δW_{i+7} hold with probability one. Consider, for example, δW_{i+1}. From (2), we have

$$\begin{aligned}
\delta W_{i+1} &= -1 - \delta\Sigma_1^i(1) - \delta f_{IF}^i(1,0,0) \\
&= x - (\Sigma_1(e_i+1) - \Sigma_1(e_i)) - (f_{IF}(e_i+1, e_{i-1}, e_{i-2}) - f_{IF}(e_i, e_{i-1}, e_{i-2})) \\
&= -1 - (0 - (-1)) - (e_{i-2} - e_{i-1}) \\
&= -2 - \gamma + \gamma + 1 \\
&= -1.
\end{aligned}$$

Similarly, conditions on δW_{i+5}, δW_{i+6} and δW_{i+7} can be verified. Conditions on δW_{i+2}, δW_{i+4} and δW_{i+5} cannot be satisfied in the manner described above. A special method to satisfy these conditions is described in § 5.1 later.

3.2 Conditions on the Differential Path of Column II of Table 2

The message word differences, δW_{i+1} and δW_{i+3} are computed from the following equations:

$$\delta W_{i+1} = -1 - \delta f_{IF}^i(1,0,0) - \delta\Sigma_1(1), \tag{7}$$

$$\delta W_{i+3} = -\delta f_{IF}^{i+2}(-1,-1,1) - \delta\Sigma_1(-1). \tag{8}$$

Intermediate registers need to satisfy the following conditions:

$$\begin{aligned} &a_{i-3} = -2, a_{i-2} = a_{i-1} = a_i = -1,\ a_{i+1} = a_{i+2} = 0,\\ &e_{i+1} = 0, e_{i+2} = 0, e_{i+3} = e_{i+4} = e_{i+5} = e_{i+6} = -1,\\ &e_i - e_{i-1} + 1 = 0. \end{aligned} \tag{9}$$

All the conditions in (9) can be deterministically satisfied by choosing message words carefully. This ensures the success probability of 1 for this local collision. These conditions can be derived in the same way as in [5].

Satisfying the Conditions. Note that, using (22) in § A, the message word W_k can be chosen to set either a_k or e_k to a desired value. Therefore the conditions on a_{i-3}, a_{i-2}, a_{i-1}, a_i, a_{i+2}, e_{i+3}, e_{i+4}, e_{i+5} and e_{i+6} can be satisfied deterministically. After this, we are left with conditions on e_{i+1}, e_{i+2} and e_i only. Two out of these three conditions are satisfied automatically as shown next.

From (1), we get:

$$\begin{aligned} e_{i+1} &= a_{i-3} + a_{i+1} - \Sigma_0(a_i) - f_{MAJ}(a_i, a_{i-1}, a_{i-2})\\ &= -2 + 0 - \Sigma_0(-1) - f_{MAJ}(-1,-1,-1)\\ &= 0. \end{aligned}$$

Similarly, we get $e_{i+2} = 0$. Now we consider the last remaining condition $e_i - e_{i-1} + 1 = 0$. Using (1), we get:

$$\begin{aligned} 0 &= 1 + e_i - e_{i-1}\\ &= 1 + (a_{i-4} + a_i - \Sigma_0(a_{i-1}) - f_{MAJ}(a_{i-1}, a_{i-2}, a_{i-3}))\\ &\quad -(a_{i-5} + a_{i-1} - \Sigma_0(a_{i-2}) - f_{MAJ}(a_{i-2}, a_{i-3}, a_{i-4}))\\ &= a_{i-4} - a_{i-5} + 2 + f_{MAJ}(-1,-2,a_{i-4}). \end{aligned}$$

This implies,

$$a_{i-5} = a_{i-4} + 2 + f_{MAJ}(-1,-2,a_{i-4}). \tag{10}$$

Equation (10) defines the register value a_{i-5} in terms of the register value a_{i-4}. But a_{i-4} will be computed only after a_{i-5} is available. To resolve this, we first choose any arbitrary value for a_{i-4} and then compute the required value of a_{i-5}. From (22), we can ensure the deterministic success of the required condition using the free words W_{i-5} and W_{i-4}.

4 The Deterministic 22-step SHA-2 Attack

In [5], a single local collision spanning from Step 6 to Step 14 is used and a 21-step collision for SHA-256 is obtained probabilistically. We use a similar method for our attack but this time we use the local collision of Table 2 spanning from Step 7 to Step 15. Message words are given by Column (II). The SHA-2 design has freedom of message words W_0 to W_{15}. Since the local collision spans this range only, we can deterministically satisfy all the conditions from (9). The message words after Step 16 are generated by message expansion. The local collision is chosen in such a way that the message expansion produces no difference in words W_i and W_i' for $i \in \{16, 17, \ldots 21\}$. This results in a deterministic 22-step attack. We explain this fact below.

First of all, note that the local collision starts from Step 7. It can be seen from the structure of the local collision that $\delta W_7 = 1$ and $\delta W_9 = \delta W_{11} = \delta W_{12} = \delta W_{13} = \delta W_{14} = 0$. In addition, δW_{15}

is -1. Messages outside the span of the local collision are taken to have zero differentials. Therefore $\delta W_i = 0$ for $i \in \{0, 1, 2, 3, 4, 5, 6\}$. Consider the first 6 steps of message expansion for SHA-2 next.

$$\left.\begin{aligned} W_{16} &= \sigma_1(W_{14}) + \ W_9 + \sigma_0(W_1) + W_0, \\ W_{17} &= \underline{\sigma_1(W_{15}) + W_{10}} + \sigma_0(W_2) + W_1, \\ W_{18} &= \sigma_1(W_{16}) + W_{11} + \sigma_0(W_3) + W_2, \\ W_{19} &= \underline{\sigma_1(W_{17})} + W_{12} + \sigma_0(W_4) + W_3, \\ W_{20} &= \sigma_1(W_{18}) + W_{13} + \sigma_0(W_5) + W_4, \\ W_{21} &= \underline{\sigma_1(W_{19})} + W_{14} + \sigma_0(W_6) + W_5. \end{aligned}\right\} \tag{11}$$

Terms which *may have* non-zero differentials in the above equations are underlined. To obtain 22-step collisions in SHA-2, it is sufficient to ensure that $\delta\{\sigma_1(W_{15}) + W_{10}\} = 0$ so that $\delta W_{17} = 0$. This also ensures that next 4 steps of the message expansion do not produce any difference, and we have a 22-step collision.

4.1 Ensuring $\delta\{\sigma_1(W_{15}) + W_{10}\} = 0$

Since $\delta W_{15} = -1$, we need the following condition to be satisfied:

$$\sigma_1(W_{15}) - \sigma_1(W_{15} - 1) = \delta W_{10}. \tag{12}$$

The local collision defines the register values as per (9). The message word difference δW_{10} is defined by (8). Simplifying this expression (where the starting step $i = 7$), we get:

$$\begin{aligned} \delta W_{10} &= -\delta f^9_{IF}(-1, -1, 1) - \delta\Sigma_1(-1) \\ &= -f_{IF}(e_9 - 1, f_9 - 1, g_9 + 1) + f_{IF}(e_9, f_9, g_9) - \Sigma_1(e_9 - 1) + \Sigma_1(e_9) \\ &= -f_{IF}(e_9 - 1, e_8 - 1, e_7 + 1) + f_{IF}(e_9, e_8, e_7) - \Sigma_1(e_9 - 1) + \Sigma_1(e_9) \\ &= -f_{IF}(-1, -1, e_7 + 1) + f_{IF}(0, 0, e_7) - \Sigma_1(-1) + \Sigma_1(0) \\ &= -(-1) + e_7 - (-1) + 0 \\ &= e_7 + 2. \end{aligned}$$

Using the CDE, we can express the right hand side of the above expression as:

$$\begin{aligned} e_7 + 2 &= a_3 + a_7 - \Sigma_0(a_6) - f_{MAJ}(a_6, a_5, a_4) + 2 \\ &= a_3 + (-1) - \Sigma_0(-1) - f_{MAJ}(-1, -1, -2) + 2 \\ &= a_3 - 1 - (-1) - (-1) + 2 \\ &= a_3 + 3. \end{aligned}$$

In satisfying (10), we needed to choose any arbitrary value of $a_{i-4} = a_3$ first. The above analysis implies that we can deterministically satisfy $\delta(\sigma_1(W_{15}) + W_{10})$ as follows:

1. Choose any arbitrary value for W_{15}. This defines the difference $\sigma_1(W_{15}) - \sigma_1(W_{15} - 1)$. Let it be called "`DELTA`".
2. From (12), δW_{10} must take the value `DELTA`. This can be obtained by setting $a_3 =$ `DELTA` $- 3$.

4.2 Algorithm to Obtain 22-step Collisions

Note that (22) in § A is used at Step i of the hash evaluation. Registers $(a_{i-1}, b_{i-1}, \ldots h_{i-1})$ are available at this step and the output register a_i or e_i can be controlled by selecting W_i suitably. For instance, if we wish to make a_i to be zero, then we can calculate the suitable value of W_i from (22) which will make this happen. We define two functions which return the required message word W_i to set the register value a_i or e_i to desired values, say `desired_a` and `desired_e`, at Step i. Equation 22 provides the definitions of these two functions.

1. `W_to_set_register_A(Step` i`, desired_a, Current State` $\{a_{i-1}, b_{i-1}, \ldots h_{i-1}\}$`) :`
   $= (\texttt{desired\_a} - \Sigma_0(a_{i-1}) - f_{MAJ}(a_{i-1}, b_{i-1}, c_{i-1}) - \Sigma_1(e_{i-1}) - f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) - h_{i-1} - K_i)$
2. `W_to_set_register_E(Step` i`, desired_e, Current State` $\{a_{i-1}, b_{i-1}, \ldots h_{i-1}\}$`) :`
   $= (\texttt{desired\_e} - d_{i-1} - \Sigma_1(e_{i-1}) - f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) - h_{i-1} - K_i)$

The algorithm to obtain message pairs leading to deterministic 22-step collisions for SHA-2 family in described in Table 15.

5 A General Idea for Obtaining 23 and 24-Step SHA-2 Collisions

Obtaining deterministic collisions up to 22 steps did not require the (single) local collision to extend beyond step 15. For obtaining collisions for more number of steps, we will need to start the local collision at Step 8 (or farther) and hence the local collision will end at Step 16 (or farther). This will require us to analyze the message expansion more carefully.

For obtaining collisions up to 22 steps, we also needed to consider message expansion. But, following Nikolić and Biryukov, we ensured that there were no differences in message words from Step 16 onwards. However, now that we consider the local collision to end at Step 16 (or farther), this will necessarily mean that one or more δW_i (for $i \geq 16$) will be non-zero. This will require a modification of the Nikolić-Biryukov strategy. Instead of requiring $\delta W_i = 0$ for $i \geq 16$, we will require $\delta W_i = 0$ for a few i's after the local collision ends. So, supposing that the local collision ends at Step 16 and we want a 23-step collision, then δW_{16} is necessarily -1 and we will require $\delta W_{17} = \cdots = \delta W_{22} = 0$.

5.1 Satisfying Conditions on the Differential Path

In § 3.1, we saw that some of the conditions on δW's of Column (I) of Table 2 can be satisfied easily. However, conditions on δW_{i+2}, δW_{i+3} and δW_{i+4} give rise to the following conditions on the values of λ, γ and μ.

$$\left.\begin{aligned} \delta W_{i+2} = \delta_1 &= -1 - \Sigma_1(\mu - 1) + \Sigma_1(\mu) - f_{IF}(\mu - 1, 0, \gamma + 1) + f_{IF}(\mu, -1, \gamma + 1) \\ \delta W_{i+3} = \delta_2 &= -\Sigma_1(\lambda - 1) + \Sigma_1(\lambda) - f_{IF}(\lambda - 1, \mu - 1, 0) + f_{IF}(\lambda, \mu, -1) \\ 1 &= -f_{IF}(\lambda - 1, \lambda - 1, \mu - 1) + f_{IF}(\lambda - 1, \lambda, \mu). \end{aligned}\right\} \quad (13)$$

Similar equations for the Nikolić-Biryukov differential path have been reported in [2] and a method for solving them has been discussed. The method to solve these equation is different for SHA-256 and for SHA-512. We discuss the exact details about solving them later. In describing our attacks on the SHA-2 family, we assume that some solutions to these equations have been obtained. These solutions are required to obtain colliding message pairs for the hash functions.

6 23-Step SHA-2 Collisions

We show that by suitably placing a local collision of the type described in Column (I) of Table 2 and using proper values for α, γ and μ, it is possible to obtain 23-step collisions for SHA-2.

6.1 Case $i = 8$

The local collision is started at $i = 8$ and ends at $i = 16$. Setting $\beta = \overline{\alpha}$, $u = 0$ and $\delta_1 = 0$, we need to choose a suitable value for δ_2 which is the value of $\delta W_{i+3} = \delta W_{11}$. For this case, we let $\delta = \delta_2$.

Since the local collision ends at Step 16, it necessarily follows that $\delta W_{16} = -1$. Consequently, we need to consider δW_{18} to ensure that it is zero. Since the collision starts at $i = 8$, all δW_j for $0 \leq j \leq 7$ are zero. Consequently, we can write $\delta W_{18} = \delta\sigma_1(W_{16}) + \delta W_{11}$, where $\delta\sigma_1(W_{16}) =$

$\sigma_1(W_{16}-1)-\sigma_1(W_{16})$. So, for δW_{18} to be zero, we need $\delta W_{11} = -\delta\sigma_1(W_{16})$, so that δW_{11} should be one of the values which occur in the distribution of $\sigma_1(W)-\sigma_1(W-1)$ for some W.

Obtaining proper values for the constants only ensures that the local collision holds from Steps i to $i+8$ as expected. It does not, however, guarantee that the reduced round collision holds. In the present case, we need to have δW_{18} to be zero. This will happen only if W_{16} takes a value such that $\sigma_1(W_{16}-1)-\sigma_1(W_{16})$ is equal to $-\delta$. This can be ensured probabilistically in the following manner. Let the frequency of δ used in the attack be freq_δ. This means that trying approximately freq_δ possible random choices of W_0 and W_1, we expect a proper value of W_{16} and hence, a 23-step collision for SHA-2. We discuss the cases of SHA-256 and SHA-512 separately later.

Since $i=8$, from Table 3, we see that a_6 to a_{10} get defined and e_6 to e_{14} get defined. Using CDE, the values of e_9 down to e_6 is set by fixing values of a_5 down to a_2. In other words, the values of a_2 to a_{10} are fixed. Now, consider

$$e_{14} = \Sigma_1(e_{13}) + f_{IF}(e_{13}, e_{12}, e_{11}) + a_{10} + e_{10} + K_{14} + W_{14}.$$

Note that in this equation all values other than W_{14} have already been fixed. So, W_{14} and hence $\sigma_1(W_{14})$ is also fixed. Now, from the update function of the a register, we can write

$$W_9 = a_9 - \Sigma_0(a_8) - f_{MAJ}(a_8, a_7, a_6) - \Sigma_1(e_8) - f_{IF}(e_8, e_7, e_6) - e_5 - K_9.$$

On the right hand side, all quantities other than e_5 have fixed values. Using CDE,

$$e_5 = a_5 + a_1 - \Sigma_0(a_4) - f_{MAJ}(a_4, a_3, a_2).$$

Again in the right hand side, all quantities other than a_1 have fixed values. So, we can write $W_9 = C - a_1$, where C is a fixed value. (This relation has already been observed in [2].)

Now,

$$a_1 = \Sigma_0(a_0) + f_{MAJ}(a_0, b_0, c_0) + \Sigma_1(e_0) + f_{IF}(e_0, f_0, g_0) + h_0 + K_1 + W_1$$

where a_0 and e_0 depend on W_0 whereas b_0, c_0, f_0, g_0 and h_0 depend only on IV and hence are constants. Thus, we can write $a_1 = \Phi(W_0) + W_1$, where

$$\Phi(W_0) = \Sigma_0(a_0) + f_{MAJ}(a_0, b_0, c_0) + \Sigma_1(e_0) + f_{IF}(e_0, f_0, g_0) + h_0 + K_1.$$

We write $\Phi(W_0)$ to emphasize that this depends only on W_0. At this point, we can write

$$\begin{aligned} W_{16} &= \sigma_1(W_{14}) + W_9 + \sigma_0(W_1) + W_0 \\ &= \sigma_1(W_{14}) + C - \Phi(W_0) - W_1 + \sigma_0(W_1) + W_0 \\ &= D - \Phi(W_0) - W_1 + \sigma_0(W_1) + W_0. \end{aligned}$$

Estimate of Computation Effort. Let there be freq_δ values of W_{16} for which $\sigma(W_{16}-1)-\sigma(W_{16})$ equals δ. So, we have to solve this equation for W_0 and W_1 such that W_{16} is one of these freq_δ possible values. The simplest way to do this is to try out random choices of W_0 and W_1 until W_{16} takes one of the desired values. On an average, success is obtained after freq_δ trials. Each trial corresponds to about a single step of SHA-2 computation. So, the total cost of finding suitable W_0 and W_1 is about $\frac{\mathsf{freq}_\delta}{2^{4.5}}$ tries of 23-step SHA-2 computations.

For each such solution (W_0, W_1) and an arbitrary choice of W_{15} we obtain a 23-step collision for SHA-2. Note that after W_0 and W_1 has been obtained everything else is deterministic, i.e., no random tries are required. The task of obtaining a suitable W_0 and W_1 can be viewed as a pre-computation of the type required to find the values of α, γ and μ. Then, the actual task of finding collisions becomes deterministic.

6.2 Case $i = 9$

It is possible to place the local collision from Step 9 to Step 17 and then perform an analysis to show that it is possible to obtain 23-step collisions for the Sanadhya-Sarkar differential path. We do not provide these details, since essentially the same technique with an additional constraint is required for 24-step collision for which we provide complete details.

6.3 Relation to the 23-Step Collision from [2]

The NB local collision has been used in [2]. The local collision was placed from Step 9 to Step 17. In comparison, we have shown that the SS local collision gives rise to two kinds of 23-step collision. The first one is obtained by placing the local collision from Steps 8 to 16, and the second one is obtained by placing the local collision from Steps 9 to 17.

The description of the attack in [2] is quite complicated. First they consider a 23-step pseudo-collision which is next converted into 23-step collision. This two-step procedure is unnecessary. Our analysis allows us to directly describe the attacks.

7 24-Step Collisions

The local collision described in Column (I) of Table 2 is placed from Step $i = 10$ to Step $i + 8 = 18$ with $u = 1$. The values of δ_1, δ_2 as well as suitable values of α, γ and μ need to be chosen.

Since, the collision ends at Step 18 and $u = 1$, we will have $\delta W_{17} = 1$ and $\delta W_{18} = -1$. As a result, to ensure $\delta W_{19} = \delta W_{20} = 0$, we need to have $\delta_1 = \delta W_{12} = -(\sigma_1(W_{17} + 1) - \sigma_1(W_{17}))$ and $\delta_2 = \delta W_{13} = -(\sigma_1(W_{18} - 1) - \sigma_1(W_{18}))$. Based on the differential behaviour of σ_1 described in § A.1, we should try to choose δ_1 and δ_2 such that $\mathsf{freq}_{-\delta_1}$ and freq_{δ_2} are as high as possible. (Here $-\delta_1$ denotes $-\delta_1 \bmod 2^n$, where n is the word size 32 or 64.) But, at the same time, the chosen δ_1 and δ_2 must be such that (13) are satisfied.

Now we consider Table 3. This table tells us what the values of the different a and e-registers need to be. Since messages up to W_{15} are free, we can set values for a and e registers up to Step 15. But, we see that $e_{16} = -1 - u = -2$. This can be achieved by setting W_{16} to

$$W_{16} = e_{16} - \Sigma_1(e_{15}) - f_{IF}(e_{15}, e_{14}, e_{13}) - a_{12} - e_{12} - K_{16}. \tag{14}$$

Since we want $e_{16} = -2$ and all other values on the right hand side are constants, we have that W_{16} is a constant value. On the other hand, W_{16} is defined by message recursion. So, we have to ensure that W_{16} takes the correct value. In addition, we need to ensure that W_{17} and W_{18} take values such that $\sigma_1(W_{17} + 1) - \sigma_1(W_{17}) = -\delta_1$ and $\sigma_1(W_{18} - 1) - \sigma_1(W_{18}) = -\delta_2$.

Since $i = 10$, from Table 3, we see that a_8 to a_{12} have to be set to fixed values and e_8 to e_{16} have to be set to fixed values. Using CDE, the values of e_{11} down to e_8 are determined by a_7 to a_4. So, the values of a_0 to a_3 are free and correspondingly the choices of words W_0 to W_3 are free.

We have already seen that W_{16} is a fixed value. Note that

$$\left.\begin{aligned} W_{14} &= e_{14} - \Sigma_1(e_{13}) - f_{IF}(e_{13}, e_{12}, e_{11}) - a_{10} - e_{10} - K_{14} \\ W_{15} &= e_{15} - \Sigma_1(e_{14}) - f_{IF}(e_{14}, e_{13}, e_{12}) - a_{11} - e_{11} - K_{15}. \end{aligned}\right\} \tag{15}$$

Since for both equations, all the quantities on the right hand side are fixed values, so are W_{14} and W_{15}.

Using CDE twice, we can write

$$\left.\begin{aligned} W_9 &= -W_1 + C_4 + f_{MAJ}(a_4, a_3, a_2) - \Phi_0 \\ W_{10} &= -W_2 + C_5 + f_{MAJ}(a_5, a_4, a_3) - \Phi_1 \\ W_{11} &= -W_3 + C_6 + f_{MAJ}(a_6, a_5, a_4) - \Phi_2 \end{aligned}\right\} \tag{16}$$

where

$$\left.\begin{array}{l} C_i = e_{i+5} - \Sigma_1(e_{i+4}) - f_{IF}(e_{i+4}, e_{i+3}, e_{i+2}) - 2a_{i+1} - K_{i+5} + \Sigma_0(a_i) \\ \Phi_i = \Sigma_0(a_i) + f_{MAJ}(a_i, b_i, c_i) + \Sigma_1(e_i) + f_{IF}(e_i, f_i, g_i) + h_i + K_{i+1}. \end{array}\right\} \quad (17)$$

Using the expressions for W_9, W_{10} and W_{11} we obtain the following expressions for W_{16}, W_{17} and W_{18}.

$$\left.\begin{array}{l} W_{16} = \sigma_1(W_{14}) + C_4 - W_1 + f_{MAJ}(a_4, a_3, a_2) - \Phi_0 + \sigma_0(W_1) + W_0 \\ W_{17} = \sigma_1(W_{15}) + C_5 - W_2 + f_{MAJ}(a_5, a_4, a_3) - \Phi_1 + \sigma_0(W_2) + W_1 \\ W_{18} = \sigma_1(W_{16}) + C_6 - W_3 + f_{MAJ}(a_6, a_5, a_4) - \Phi_2 + \sigma_0(W_3) + W_2. \end{array}\right\} \quad (18)$$

We need to ensure that W_{16} has the desired value given by (14) and that W_{17} and W_{18} take values which lead to desired values for $\delta\sigma_1(W_{17})$ and $\delta\sigma_1(W_{18})$ as explained above.

The only free quantities are W_0 to W_3 which determine a_0 to a_3. The value of C_4 depends on e_8, e_7 and e_6, where e_8 has a fixed value and e_7 and e_6 are in turn determined using CDE by a_3 and a_2. Similarly, C_5 is determined by e_9, e_8 and e_7; where e_9, e_8 have fixed values and e_7 is determined using a_3. The value of C_6 on the other hand is fixed. Coming to the Φ values, Φ_0 is determined only by W_0; Φ_1 determined by W_0 and W_1; and Φ_2 determined by W_0, W_1 and W_2. Let

$$D = W_{16} - (\sigma_1(W_{14}) + C_4 + f_{MAJ}(a_4, a_3, a_2) - \Phi_0 + W_0). \quad (19)$$

If we fix W_0 and a_3, a_2, then the value of D gets fixed and we need to find W_1 such that the following equation holds.

$$D = -W_1 + \sigma_0(W_1). \quad (20)$$

A guess-then-determine algorithm can be used to solve this equation. This algorithm will be different for SHA-256 and for SHA-512 since the σ_0 function is different for the two. The guess-then-determine algorithms for both SHA-256 and SHA-512 are described in § B.

Solving (20) Using Table Look-Up. An alternative approach would be to use a pre-computed table. For each of the 2^n possible W_1s (n is the word size 32 or 64), prepare a table of entries $(W_1, -W_1 + \sigma_0(W_1))$ sorted on the second column. Then all solutions (if there are any) for (20) can be found by a simple look-up into the table using D. The table would have 2^n entries and if a proper index structure is used, then the look-up can be done very fast. We have not implemented this method.

Given $a_1, b_1, \ldots, h_1$ and a_2 the value of W_2 gets uniquely defined; similarly, given $a_2, b_2, \ldots, h_2$ and a_3, the value of W_3 gets uniquely defined. The equations are the following.

$$\left.\begin{array}{l} W_2 = a_2 - (\Sigma_0(a_1) + f_{MAJ}(a_1, b_1, c_1) + h_1 + \Sigma_1(e_1) + f_{IF}(e_1, f_1, g_1) + K_2) \\ W_3 = a_3 - (\Sigma_0(a_2) + f_{MAJ}(a_2, b_2, c_2) + h_2 + \Sigma_1(e_2) + f_{IF}(e_2, f_2, g_2) + K_3) \end{array}\right\} \quad (21)$$

The strategy for determining suitable $W_0, \ldots, W_3$ is the following.

1. Make random choices for W_0 and a_2, a_3.
2. Run SHA-2 with W_0 and determine Φ_0.
3. From a_3 and a_2 determine e_7 and e_6 using CDE.
4. Determine C_4 using (17) and then D using (19).
5. Solve (20) for W_1 using the guess-then-determine algorithm.
6. Run SHA-2 with W_1 to define $a_1, \ldots, h_1$.
7. Determine Φ_1 using (17) and then W_2 using (21).
8. Run SHA-2 with W_2 to define $a_2, \ldots, h_2$.
9. Determine Φ_2 using (17) and then W_3 using (21).
10. Compute W_{17} and W_{18} using (18).
11. If $\sigma_1(W_{17} + 1) - \sigma_1(W_{17}) = -\delta_1$ and $\sigma_1(W_{18} - 1) - \sigma_1(W_{18}) = \delta_2$, then return W_0, W_1, W_2 and W_3.

The values of W_0, W_1, W_2 and W_3 returned by this procedure ensure that the local collision ends properly at Step 18 and that $\delta W_j = 0$ for $j = 19, \ldots, 23$. This provides a 24-step collision.

Estimate of Computation Effort. Let Step 5 involve a computation of g operations, where each operation is much faster than a single step of SHA-2; by our assessment the time for each operation is around 2^{-4} times the cost of a single step of SHA-2. Thus, the time for Step 5 is about $\frac{g}{2^4}$ single SHA-2 steps. Further, let the success probability of the guess-then-determine attack be p. Then Step 5 needs to be repeated roughly $\frac{1}{p}$ times to obtain a solution.

By the choice of δ_1, the equality $\sigma_1(W_{17}+1) - \sigma_1(W_{17}) = -\delta_1$ holds roughly with probability $\frac{\mathsf{freq}_{\delta_1}}{2^n}$ while by the choice of δ_2 the equality $\sigma_1(W_{18}-1) - \sigma_1(W_{18}) = \delta_2$ holds roughly with probability $\frac{\mathsf{freq}_{\delta_2}}{2^n}$ and we obtain success in Step 11 with roughly $\frac{\mathsf{freq}_{\delta_1} \times \mathsf{freq}_{\delta_2}}{2^{2n}}$ probability. So, the entire procedure needs to be carried out around $\frac{2^{2n}}{\mathsf{freq}_{\delta_1} \times \mathsf{freq}_{\delta_2}}$ times to obtain a collision.

The guess-then-determine step takes about $g/2^4$ single SHA-2 steps. The time for executing the entire procedure once is about $(\frac{g}{2^4} + 3)$ single SHA-2 steps which is about $2^{-4.5} \times (\frac{g}{2^4} + 3)$ 24-step SHA-2 computations. Since the entire process needs to be repeated many times for obtaining success, the number of 24-step SHA-2 computations till success is obtained is about $(\frac{2^{2n}}{\mathsf{freq}_{\delta_1} \times \mathsf{freq}_{\delta_2}}) \times (2^{-4.5} \times (\frac{g}{2^4} + 3) \times \frac{1}{p})$.

If (20) is solved using a table look-up, then the cost estimate changes quite a lot. The cost of Step 5 reduces to about a single SHA-2 step so that the overall cost reduces to about $(\frac{2^{2n}}{\mathsf{freq}_{\delta_1} \times \mathsf{freq}_{\delta_2}}) \times (2^{-4.5} \times 3 \times \frac{1}{p})$ 24-step SHA-2 computations. The trade-off is that we need to use a look-up table having 2^n entries.

8 Exhibiting Colliding Message Pairs

The first step in producing colliding pair of messages for different number of rounds is to solve equations (13). For the case of the Sanadhya-Sarkar differential path, the following strategy (which is somewhat similar to the case given in [2]) can be used to solve equations (13).

- The third equation holds with probability 1 if both λ and μ are odd.
- Given that λ and μ are odd, the second equation simplifies to $\delta_2 = -\Sigma_1(\lambda - 1) + \Sigma_1(\lambda) + \overline{(\lambda - 1)}$. For a given odd value of δ occurring in the distribution of $\sigma_1(W) - \sigma_1(W-1)$, it is possible to solve this equation for odd λ.
- Given such a λ, it is easy to solve the equation $\lambda = -\Sigma_0(\overline{\alpha})$ to obtain a suitable value of α, since Σ_0 is an invertible mapping for both SHA-256 and SHA-512.
- For the first equation, the term $-f_{IF}(\mu - 1, 0, \gamma + 1) + f_{IF}(\mu, -1, \gamma + 1)$ is equal to μ, if γ is odd. This term is equal to $\mu - 1$ if γ is even. Further, we note that $-\Sigma_1(\mu - 1) + \Sigma_1(\mu)$ is always even for both SHA-256 and SHA-512. Thus taking an arbitrary odd value of γ, the first equation is in the single variable μ and can be solved easily for a given δ_1.

Now we provide proofs of the observations above.

Lemma 1 *The third equation is satisfied for any odd λ and odd μ.*

Proof. We have to show that

$$1 = -f_{IF}(\lambda - 1, \lambda - 1, \mu - 1) + f_{IF}(\lambda - 1, \lambda, \mu).$$

The quantities λ and $\lambda - 1$ differ only in their least significant bit since λ is odd. Similarly, μ and $\mu - 1$ differ only in their least significant bit since μ is odd. Let x_i denote the i^{th} bit of x, then λ_0=1, $(\lambda - 1)_0 = 0$, $\mu_0 = 1$ and $(\mu - 1)_0 = 0$. Let $(\lambda - 1)_i = \lambda_i = 1$ and $(\lambda - 1)_j = \lambda_j = 0$ for some non-zero indices i and j. Also, let $\mu_i = b_1$ and $\mu_j = b_2$ for these bit positions i and j. Now we are ready to write the bit patterns of the quantities occuring in the third equation.

bit	$63 \ldots$	$i \ldots$	$j \ldots 0$
$\lambda - 1$	$. \ \ldots$	$1 \ldots$	$0 \ldots 0$
λ	$. \ \ldots$	$1 \ldots$	$0 \ldots 1$
μ	$. \ \ldots$	$b_1 \ldots$	$b_2 \ldots 1$
$f_{IF}(\lambda - 1, \lambda, \mu)$	$. \ \ldots$	$1 \ldots$	$b_2 \ldots 1$

Similarly,

bit	$63 \ldots$	$i \ldots$	$j \ldots 0$
$\lambda - 1$	$. \ \ldots$	$1 \ldots$	$0 \ldots 0$
$\lambda - 1$	$. \ \ldots$	$1 \ldots$	$0 \ldots 0$
$\mu - 1$	$. \ \ldots$	$b_1 \ldots$	$b_2 \ldots 0$
$f_{IF}(\lambda - 1, \lambda - 1, \mu - 1)$	$. \ \ldots$	$1 \ldots$	$b_2 \ldots 0$

From the two bit patterns above, we get that

$$f_{IF}(\lambda - 1, \lambda, \mu) - f_{IF}(\lambda - 1, \lambda - 1, \mu - 1) = 1.$$

□

Lemma 2 *For odd λ and odd μ, the second equation simplifies to $\delta_2 = -\Sigma_1(\lambda - 1) + \Sigma_1(\lambda) + \overline{(\lambda - 1)}$.*

Proof. Consider the following expression

$$-f_{IF}(\lambda - 1, \mu - 1, 0) + f_{IF}(\lambda, \mu, -1).$$

Similar to the proof of the previous lemma, we consider the bit patterns of the quantities occuring in the above equation. Let $\lambda_i = 1$ and $\lambda_j = 0$ for some non-zero i, j. Also, let $\mu_i = b_1$ and $\mu_j = b_2$. Then the following bit patterns can be seen for the various quantities.

bit	$63 \ldots$	$i \ldots$	$j \ldots 0$
λ	$. \ \ldots$	$1 \ldots$	$0 \ldots 1$
μ	$. \ \ldots$	$b_1 \ldots$	$b_2 \ldots 1$
-1	$1 \ \ldots$	$1 \ldots$	$1 \ldots 1$
$f_{IF}(\lambda, \mu, -1)$	$. \ \ldots$	$b_1 \ldots$	$1 \ldots 1$

Similarly,

bit	$63 \ldots$	$i \ldots$	$j \ldots 0$
$\lambda - 1$	$. \ \ldots$	$1 \ldots$	$0 \ldots 0$
$\mu - 1$	$. \ \ldots$	$b_1 \ldots$	$b_2 \ldots 0$
0	$0 \ \ldots$	$0 \ldots$	$0 \ldots 0$
$f_{IF}(\lambda - 1, \mu - 1, 0)$	$. \ \ldots$	$b_1 \ldots$	$0 \ldots 0$

From the two bit patterns above, we get that $f_{IF}(\lambda, \mu, -1)$ and $f_{IF}(\lambda - 1, \mu - 1, 0)$ will have the same bit value whenever the corresponding bit of λ is 1 and different bit value whenever the corresponding bit of λ is 0, except the least significant bit which will always be different. Comparing this difference with the bit pattern $\overline{\lambda - 1}$, we obtain

$$f_{IF}(\lambda, \mu, -1) - f_{IF}(\lambda - 1, \mu - 1, 0) = \overline{\lambda - 1}.$$

This completes the proof.

□

Lemma 3 *For odd μ and odd γ, the first equation simplifies to $\delta_1 = -1 - \Sigma_1(\mu - 1) + \Sigma_1(\mu) + \mu$.*

Proof. By considering the bit patters of μ, $\mu - 1$ and $\gamma + 1$ the following can be proved in a manner similar to the previous two lemmas.

$$f_{IF}(\mu, -1, \gamma + 1) - f_{IF}(\mu - 1, 0, \gamma + 1) = \begin{cases} \mu & \text{if } \gamma \text{ is odd.} \\ \mu - 1 & \text{if } \gamma \text{ is even.} \end{cases}$$

Substituting the above value in the equation for δ_1 gives the required proof.

□

8.1 SHA-256

- For SHA-256 we did not solve the second equation explicitly since random search is itself good enough, producing a solution in few seconds. The maximum value of freq_δ for odd δ is 2^{16}. One example is $\delta =$ `ff006001` as shown in Table 6.
- Given such a λ, we obtain $\alpha = \overline{\Sigma_0^{-1}(-\lambda)}$.
- Now, we choose any arbitrary odd γ and solve for an odd μ such that the first equation holds.

Solving all the three equations for α, γ and μ can be done in a few seconds on a current PC for step reduced SHA-256. Examples are provided in Table 4.

Table 4. Values leading to collisions for different number of steps of SHA-256. The value of i denotes the start point of the local collision, i.e., the local collision is placed from Step i to $i + 8$.

(# rnds, i)	δ_1	δ_2	u	α	λ	γ	μ
(23, 8)	0	`ff006001`	0	`32b308b2`	`051f9f7f`	`684e62b7`	`041fff81`
(23, 9) (24, 10)	`00006000`	`ff006001`	1	`32b308b2`	`051f9f7f`	`98e3923b`	`fbe05f81`

23-step Collision. The $\delta = \delta_2$ that we have used (shown in Table 4) is such that $\mathsf{freq}_\delta = 2^{16}$ and so by trying approximately 2^{16} possible random choices of W_0 and W_1 we expect a proper value for W_{16} and hence, a 23-step collision. Following the analysis of computation effort in § 6.1, the effort required is about $\frac{\mathsf{freq}_\delta}{2^{4.5}} = \frac{2^{16}}{2^{4.5}} = 2^{11.5}$ trials of 23-step SHA-256. In [2], the corresponding probability is 2^{-19} and the computaional effort is $2^{14.5}$ trials. So, our technique is an improvement. A message pair colliding for 23-step SHA-256 is given in § C.

24-step Collision. As mentioned in § A.1, if we choose δ_2 such that $\mathsf{freq}_{\delta_2} > 2^{16}$, then it is not possible to solve (13). So we choose $\delta_2 =$ `ff006001` with $\mathsf{freq}_{\delta_2} = 2^{16}$. Also, we choose $\delta_1 =$ `00006000` so that $-\delta_1 =$ `ffffa000` and $\mathsf{freq}_{-\delta_1} = 2^{29} + 2^{26}$. For these values of δ_1 and δ_2, it is possible to solve (13) to obtain suitable α, γ and μ, which in turn determine $\beta = \overline{\alpha}$ and λ. An example of these values is shown in Table 4. The same values also hold for obtaining 23-step collision by placing a local collision from Step 9 to 17. Message pair colliding for 24-step SHA-256 is given in § C.

Guess-Then-Determine Algorithm. In § 7, it was mentioned that a guess-then-determine algorithm is used to solve (20). We discuss this algorithm for SHA-256. By guessing a total of 18 bits (15 least significant bits of W_1 and three other possible carry bits), it is possible to reconstruct the entire W_1 and then determine whether the reconstructed value is correct. Thus, by trying a total of 2^{18} combinations, it is possible to determine whether (20) has a solution and if so to find all possible solutions. The algorithm is given in § B. (We note that in [2], it has been remarked that "*by guessing*

the least 15 bits of W_1 *the entire* W_1 *can be reconstructed and with probability* 2^{-14} *it is going to be correct*". No details are provided. In particular, the guess-then-determine algorithm that we provide in § B is not present in [2].)

In our experiments with SHA-256, we found that for almost every other value of D, (20) has solutions, the number of solutions being one or two. So, for a random choice of D, we consider (20) to hold with probability $p \approx 1$.

Complexities of The Attacks. As already mentioned, our 23-step SHA-256 attack succeeds with probability 2^{-16}, i.e., about 2^{16} random trials are required to obtain colliding message pairs. This corresponds to roughly $2^{11.5}$ 23-step SHA-256 computations.

For the 24-step attack, the values of g, freq_{δ_1} and freq_{δ_2} are 2^{18}, 2^{29} and 2^{16} respectively. From the cost analysis done in § 7, we obtain success in about $2^{28.5}$ 24-step SHA-256 computations. In our experiments, we found that the computation effort required to find $W_0, \ldots, W_3$ actually turns out to be less than the estimated effort of $2^{28.5}$ 24-step SHA-256 computations. The value of $2^{28.5}$ matches the figure given in [2], but [2] does not provide the detailed analysis of their cost.

As already explained in § 7, if (20) is solved using a table look-up, then the cost reduces to about $2^{15.5}$ 24-step SHA-256 computations. This constitutes an improved attack on 24-step SHA-256.

8.2 SHA-512

As in the case of SHA-256, we first need to solve (13) to obtain values of α, λ, γ and μ. For the 23-step attack, these equations are solved for $\delta_1 = 0$ and δ_2 being one of the high frequency values in the distribution $\sigma_1(W) - \sigma_1(W-1)$. For the 24-step attack, we need some frequently occurring δ_1 and δ_2 values. This distribution can not be computed completely as the word size is 64 bits in this case. However, a smaller distribution (of size much lesser than 2^{64} entries) can be constructed by randomly selecting W's and tabulating $\sigma_1(W) - \sigma_1(W-1)$. This distribution can then be extrapolated to obtain approximate frequencies of the actual distribution. This idea is mentioned in [2]. More details about the process of estimation of the frequencies is provided in § A.1.

It is possible to solve (13) for SHA-512 as well, although we require a slighly different approach than SHA-256. The main difference is in solving the second equation. We describe the method to solve this equation with the aid of an example. Suitables values of the constants for reduced SHA-512 attacks are given in Table 5.

Table 5. Values leading to collisions for different number of steps of SHA-512. The value of i denotes the start point of the local collision, i.e., the local collision is placed from Step i to $i+8$.

(# rnds, i)	δ_1	δ_2	u	α	λ	γ	μ
(23, 8)	0	600000000237	0	7201b90f9f8df85e	3e000007ffdc9	1	43fffff800001
(23, 9) (24, 10)	200000000008	600000000237	1	7201b90f9f8df85e	3e000007ffdc9	1	45fffff800009

Solving the Second Equation For SHA-512. As remarked earlier, for odd λ the second equation simplifies to

$$\delta_2 = -\Sigma_1(\lambda - 1) + \Sigma_1(\lambda) + \overline{(\lambda - 1)}.$$

We need to get an odd λ satisfying the above equation for a given value of δ_2. Since $-\Sigma_1(\lambda-1)+\Sigma_1(\lambda)$ is always even and $\overline{(\lambda-1)}$ is odd due to our choice of odd λ, we require δ_2 to be odd. This equation can be solved by hand. We explain the method to solve this equation for $\delta_2 = 600000000237$.

First note that $\Sigma_1(x)$ is the XOR addition of 3 n-bit quantities which are rotated/shifted forms of x. If λ is odd, then λ and $\lambda - 1$ differ only in the least significant bit. Therefore, the bit patterns of $\Sigma_1(\lambda)$ and $\Sigma_1(\lambda - 1)$ will be same except at 3 bit positions. These 3 bit positions are indexed by 23, 46 and 50. By the structure of Σ_1 function and using the fact that λ is odd (i.e. $\lambda_0 = 1$), we have the following

$$b_1 = (\Sigma_1(\lambda))_{23} = \lambda_0 \oplus \lambda_{37} \oplus \lambda_{41} = 1 \oplus \lambda_{37} \oplus \lambda_{41},$$
$$b_2 = (\Sigma_1(\lambda))_{46} = \lambda_0 \oplus \lambda_{23} \oplus \lambda_{60} = 1 \oplus \lambda_{23} \oplus \lambda_{60},$$
$$b_3 = (\Sigma_1(\lambda))_{50} = \lambda_0 \oplus \lambda_4 \ \oplus \lambda_{27} = 1 \oplus \lambda_4 \ \oplus \lambda_{27}.$$

Also, because $(\lambda - 1)_0 = 0$, we have $(\Sigma_1(\lambda - 1))_{23} = \overline{b_1}$, $(\Sigma_1(\lambda - 1))_{46} = \overline{b_2}$ and $(\Sigma_1(\lambda - 1))_{60} = \overline{b_3}$.

Now consider the bit pattern of various quantities as follows.

bit	63 ... 50 ... 46 ... 23 ... 0
$A = \Sigma_1(\lambda - 1)$	.... $\overline{b_3}$ ... $\overline{b_2}$ ... $\overline{b_1}$ ... .
$B = \Sigma_1(\lambda)$	.... b_3 ... b_2 ... b_1 ... .
$A - B$	.... 1 0... 0
δ_2	....
$A - B + \delta_2$	....

We require the quantity $(A - B + \delta_2)$ to be equal to $\overline{(\lambda - 1)}$. It is clear from the bit pattern above that the lowest 23 bits (indexed from 0 to 22) of $(A - B + \delta_2)$ will be same as those of δ_2. Equating these bits to corresponding bits of $\overline{(\lambda - 1)}$, we immediately get the lowest 23 bits of λ.

Now consider the bits between 23 and 46 of $(A - B)$. It is clear that all these bits will be equal. Further, all these bits will be equal to 1 if $b_1 = 1$ due to the borrow while subtracting B from A at bit position 23. Similarly, all these bits of $(A - B)$ will be equal to 0 if $b_1 = 0$. Our choice of δ_2 has all these bits equal to zero, hence the term $(A - B + \delta_2)$ will too have all these bits equal. But since this term is equal to $\overline{(\lambda - 1)}$, all these bits of $(\lambda - 1)$ will also be equal. Finally, note that λ and $(\lambda - 1)$ differ only in the lowest bit position, hence all the bits between 23 and 46 of λ will also be equal. In particular, we will have $\lambda_{37} = \lambda_{41}$, hence we have that $b_1 = 1 \oplus \lambda_{37} \oplus \lambda_{41} = 1$.

Continuing reasoning on bit positions in this way, for any given δ_2, either we can solve for λ or determine that a solution does not exist. For $\delta_2 =$ `600000000237` we obtained the solution $\lambda =$ `3e000007ffdc9`. Note that the method explained above does not require any particular structure of the bits of δ_2. As another example, we also solved for $\delta_2 =$ `19fffffffdd9` and obtained the solution as $\lambda =$ `2200000800227`.

Note: The first equation can be solved in a similar manner for μ for a given δ_1.

23-step Collision. The $\delta = \delta_2$ used (shown in Table 5) has frequency $\mathsf{freq}_\delta \approx 2^{43}$ and so by trying approximately $\frac{2^{64}}{2^{43}} = 2^{21}$ possible random choices of W_0 and W_1 we expect a proper value for W_{16} and hence, a 23-step collision. Following the analysis of computation effort in § 6.1, the effort required is about $\frac{\mathsf{freq}_\delta}{2^{4.5}} = \frac{2^{21}}{2^{4.5}} = 2^{16.5}$ trials of 23-step SHA-512. In [2], the corresponding effort is $2^{43.9}$. Message pair colliding for 23-step SHA-512 is given in § C.

Guess-Then-Determine Algorithm For SHA-512. In § 7, a guess-then-determine algorithm was mentioned to solve (20). We discuss the case of this algorithm for SHA-512 now. By guessing a total of 15 bits (8 least significant bits of W_1 and 7 other possible carry bits), it is possible to reconstruct the entire W_1 and then determine whether the reconstructed value is correct. Thus, by trying a total of 2^{15} combinations, it is possible to determine whether (20) has a solution and if so, to find all possible solutions. The algorithm is given in § B. (We note that in [2], it has been remarked

that by guessing the least 8 bits of W_1 the entire W_1 can be reconstructed and with probability 2^{-8} it is going to be correct. No details are provided; in particular, the guess-then-determine algorithm that we provide in § B is not present in [2].)

In our experiments with SHA-512, we found that, on average, for one in every six values of D, (20) has solutions, the number of solutions being between one and four. So, for a random choice of D, we consider (20) to hold with probability $p \approx 2^{-2.5}$.

24-step Collision. We choose $\delta_2 =$ 600000000237 with $\mathsf{freq}_{\delta_2} = 2^{43}$. Also, we choose $\delta_1 =$ 200000000008 so that $\mathsf{freq}_{-\delta_1} = 2^{61.5}$. For these values of δ_1 and δ_2, it is possible to solve (13) to obtain suitable α, γ and μ, which in turn determine λ and $\beta = \overline{\alpha}$. An example of these values is shown in Table 5. The same values also hold for obtaining 23-step collision by placing a local collision from Step 9 to 17.

The guess-then-determine attack for SHA-512 case requires $g = 2^{15}$ operations, hence following the analysis of computation effort in § 7, the effort required for 24-step SHA-512 attack is about $(\frac{2^{2\times 64}}{2^{61.5}\times 2^{43}}) \times (2^{-4.5} \times (\frac{2^{15}}{2^4} + 3) \times \frac{1}{2^{-2.5}}) = 2^{32.5}$ trials of 24-step SHA-512. In [2], the corresponding effort is 2^{53} trials of 24-step SHA-512. This significant improvement in the attack complexity allows us to provide the first example of a colliding message pair for 24-step SHA-512. The message pair colliding for 24-step SHA-512 is given in § C.

Note that using a table having 2^{64} entries to solve (20) will reduce the computational effort to about $2^{22.5}$ trials of 24-step SHA-512.

9 Some Concluding Remarks

In this work we have presented a deterministic attack against 22-step SHA-2 and probabilistic attacks against 23 and 24-step SHA-2. Hopefully this work will help understand SHA-2 family better and help in devising new attacks on longer round versions of SHA-2 family.

References

1. 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.
2. Sebastiaan Indesteege, Florian Mendel, Bart Preneel, and Christian Rechberger. Collisions and other Non-Random Properties for Step-Reduced SHA-256. *Cryptology eprint Archive*, April 2008. Available at `http://eprint.iacr.org/2008/131`, Accepted in Selected Areas in Cryptography, 2008.
3. 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.
4. 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`.
5. Ivica Nikolić and Alex Biryukov. Collisions for Step-Reduced SHA-256. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, March 26-28, 2008*, volume Pre-proceedings version of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2008.
6. 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.
7. Somitra Kumar Sanadhya and Palash Sarkar. Attacking Reduced Round SHA-256. In Steven Bellovin and Rosario Gennaro, editors, *Applied Cryptography and Network Security - ACNS 2008, 6th International Conference, New York, NY, June 03-06, 2008, Proceedings*, volume 5037 of *Lecture Notes in Computer Science*. Springer, 2008.
8. Somitra Kumar Sanadhya and Palash Sarkar. Deterministic Constructions of 21-Step Collisions for the SHA-2 Hash Family. In Editors, editor, *Information Security, 11th International Conference, ISC 2008, Taipei, Taiwan, September 2008, Proceedings*, volume To be published of *Lecture Notes in Computer Science*. Springer, 2008.

9. Somitra Kumar Sanadhya and Palash Sarkar. Non-Linear Reduced Round Attacks Against SHA-2 Hash family. In Yi Mu and Willy Susilo, editors, *Information Security and Privacy - ACISP 2008, The 13th Australasian Conference, Wollongong, Australia, 7-9 July 2008, Proceedings*, volume To appear of *Lecture Notes in Computer Science*. Springer, 2008.

A Details of the SHA-2 Hash Family

Eight registers are used in the evaluation of SHA-2. The initial value in the registers is specified by an $8 \times n$ bit IV, n=32 for SHA-256 and $n = 64$ for SHA-512. 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:

$$\left.\begin{array}{l} a_i = \Sigma_0(a_{i-1}) + f_{MAJ}(a_{i-1}, b_{i-1}, c_{i-1}) + \Sigma_1(e_{i-1}) \\ \quad\quad +f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) + h_{i-1} + K_i + W_i \\ b_i = a_{i-1} \\ c_i = b_{i-1} \\ d_i = c_{i-1} \\ e_i = d_{i-1} + \Sigma_1(e_{i-1}) + f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) \\ \quad\quad +h_{i-1} + K_i + W_i \\ f_i = e_{i-1} \\ g_i = f_{i-1} \\ h_i = g_{i-1} \end{array}\right\} \quad (22)$$

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

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

For SHA-256, the functions Σ_0 and Σ_1 are defined as:

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

For SHA-512, the corresponding functions are:

$$\begin{aligned} \Sigma_0(x) &= ROTR^{28}(x) \oplus ROTR^{34}(x) \oplus ROTR^{39}(x), \\ \Sigma_1(x) &= ROTR^{14}(x) \oplus ROTR^{18}(x) \oplus ROTR^{41}(x). \end{aligned}$$

Given the message words $W_0, W_1, \ldots, W_{15}$, for $i \geq 16$, W_i is computed as follows.

$$W_i = \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} \quad (23)$$

For SHA-256, the functions σ_0 and σ_1 are defined as:

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

And for SHA-512, they are defined as:

$$\begin{aligned} \sigma_0(x) &= ROTR^1(x) \ \oplus ROTR^8(x) \ \oplus SHR^7(x), \\ \sigma_1(x) &= ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^6(x). \end{aligned}$$

A.1 Differential Properties of σ_1

The linear function σ_1 of SHA-256 used in the message expansion has very poor differential properties with respect to modular addition. Consider the distribution of $\delta = \sigma_1(W) - \sigma_1(W-1)$ as W ranges over all 2^{32} values. The highly skewed nature of the distribution of δ was first mentioned in § 7.1 in [9], where it was utilized in improving the success probability of the 21-step SHA-256 NB attack. Later, it has been independently observed in [2] that δ takes only 6181 values and there are several values of δ which occur for more than 2^{29} or more values of W.

Let freq_δ be the number of W such that $\delta = \sigma_1(W) - \sigma_1(W-1)$. It is quite easy to prepare a list of $(\delta, \mathsf{freq}_\delta)$ values. For each of the 2^{32} values of W, compute $\delta = \sigma_1(W) - \sigma_1(W-1)$. If this δ has been obtained earlier, then increment the frequency for this δ; else insert $(\delta, \mathsf{freq}_\delta = 1)$ into the list. To do this efficiently, we need a suitable index structure for searching and inserting into the list. A height balanced tree (or AVL tree) is the optimal solution; but, for the current application, a simple (data structure) hash technique is good enough and is the technique we implemented.

Some values of $(\delta, \mathsf{freq}_\delta)$ are given in Table 6. Interestingly, we have observed that if freq_δ is greater than 2^{16}, then δ is always even.

Table 6. Some examples of high frequency values of $\delta = \sigma_1(W) - \sigma_1(W-1)$ for SHA-256.

δ	freq_δ	δ	freq_δ
ffff6000	$2^{29} + 2^{26} + 2^{25}$	0000a000	$2^{29} + 2^{26} + 2^{25}$
ffffa000	$2^{29} + 2^{26}$	00006000	$2^{29} + 2^{26}$
ff006001	2^{16}	ff005fff	2^{16}

Similar distribution for SHA-512 can also be computed by generating a list of $\delta = \sigma_1(W) - \sigma_1(W-1)$ for random W's. We created a list of 2^{25} entries and extrapolated the distribution to get some high frequency values for SHA-512. The extrapolation is done in the following manner. If a particular difference δ occurs r times in 2^{25} random trials, then we expect it to have a frequency freq_δ of about $r \times 2^{64}/2^{25}$. Some of the observed and the extrapolated frequencies are shown in Table 7.

Table 7. Some examples of high frequency values of $\delta = \sigma_1(W) - \sigma_1(W-1)$ for SHA-512. The column freq_O denotes the observed frequencies among 2^{25} random trials of computing δ. The column freq_δ contains the extrapolated values of the frequencies for the complete search space of 2^{64}.

δ	freq_O	freq_δ	δ	freq_O	freq_δ
200000000008	4795491	$2^{61.5}$	8e000000003a9	22	$2^{43.5}$
ffffdfffffffff8	4793201	$2^{61.5}$	fff26000000000c9	22	$2^{43.5}$
1fffffffff8	4792982	$2^{61.5}$	600000000237	18	$2^{43.5}$

B Guess-Then-Determine Algorithm for Solving (20)

For the ease of notation, in this section we will use W instead of W_1.

B.1 For SHA-256

Consider Table 1 where the structure of W and $\sigma_0(W)$ is shown for SHA-256. We have $-W + \sigma_0(W) = D$, where $D = (d_{31}, \ldots, d_0)$ is a 32-bit constant. For $31 \geq k \geq l \geq 0$, we will use the notation $X[k, l]$ to denote bits $x_k, \ldots, x_l$ of the 32-bit quantity X.

We explain how the guess-then-determine algorithm proceeds. Suppose that we guess $W[14,0]$. Let $X = D + W$ and $Y = (W[14,0] \gg 3) \oplus (W[14,0] \gg 7)$. Then $W[25,18] = (X \oplus Y)\&(\texttt{ff})$. Having determined $W[25,18]$ we next determine $W[29,26]$ using positions 22 to 19 of Table 1. This time, however, there may have been a possible carry into the 19th bit and we need to account for that. Let c_0 be a bit. Define $X = (D \gg 19) + (W[25,18] \gg 1) + c_0$ and $Y = (W[14,0] \gg 5) \oplus (W[25,18] \gg 4)$. Then $W[29,26] = (X \oplus Y)\&(\texttt{f})$. This illustrates the general idea and can be extended to determine the other bits. Once the entire W has been determined we need to determine whether $-W + \sigma_0(W) = D$. The entire algorithm is shown in Figure 2. This algorithm involves guessing $W[14,0]$ and bits c_0, c_1, c_2,

Fig. 1. Structure of W and $\sigma_0(W)$ for SHA-256.

W	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}	w_{16}
$W \gg 3$	0	0	0	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}
$W \ggg 7$	w_6	w_5	w_4	w_3	w_2	w_1	w_0	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}
$W \ggg 18$	w_{17}	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7	w_6	w_5	w_4	w_3	w_2

W	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0
$W \gg 3$	w_{18}	w_{17}	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7	w_6	w_5	w_4	w_3
$W \ggg 7$	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7
$W \ggg 18$	w_1	w_0	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}

Fig. 2. A guess-then-determine algorithm for solving $D = -W + \sigma_0(W)$ for SHA-256.

1. Guess $W[14,0]$.
2. Let $X = D + W$ and $Y = (W[14,0] \gg 3) \oplus (W[14,0]) \gg 7$
   and set $W[25,18] = (X \oplus Y)\&(\texttt{ff})$.
3. Guess c_0.
4. Let $X = (D \gg 19) + (W[25,18] \gg 1) + c_0$ and $Y = (W[14,0] \gg 5) \oplus (W[25,18] \gg 4)$
   and set $W[29,26] = (X \oplus Y)\&(\texttt{f})$.
5. Guess c_1.
6. Let $X = (D \gg 23) + (W[25,18] \gg 6) + c_1$ and $Y = (W[14,0] \gg 9) \oplus (W[29,26] \gg 4)$
   and set $W[31,20] = (X \oplus Y)\&(\texttt{3})$.
7. Guess c_2.
8. Let $X = (D \gg 8) + (W[14,0] \gg 8) + c_2$ and $Y = (W[14,0] \gg 11) \oplus (W[29,26])$
   and set $W[31,20] = (X \oplus Y)\&(\texttt{7})$.
9. If $-W + \sigma_0(W) = D$, then output W as one solution.

which is a total of 18 bits. If the equation $D = -W + \sigma_0(W)$ does not have any solution, then none will be returned by this algorithm; on the other hand, if there is a solution or there are more than one solutions, then all solutions will be returned. A total of 2^{18} operations are required. The time for each operation is significantly less than the time for a single SHA-256 step and by our assessment it is about 2^{-4} times the time for a single SHA-256 step.

B.2 For SHA-512

Consider Table 3 where the structure of W and $\sigma_0(W)$ is shown for SHA-512. We have $-W + \sigma_0(W) = D$, where $D = (d_{63}, \ldots, d_0)$ is a 64-bit constant. For $63 \geq k \geq l \geq 0$, we will use the notation $X[k,l]$ to denote bits $x_k, \ldots, x_l$ of the 64-bit quantity X.

We explain how the guess-then-determine attack proceeds. Suppose that we guess $W[7,0]$. So we know the 7 bits $W[7,1]$ and $W[6,0]$. Now, consider the lowest 7 bits of $D + W$. We need $D + W$ to be equal to $\sigma_0(W)$. The term $\sigma_0(W)$ consists of 3 quantities XOR'ed, one of which, $W[7,1]$, is already known. The other two quantities are $W[13,7]$ and $W[14,8]$. So we can compute $X =$

$W[13,7] \oplus W[14,8] = (D+W) \oplus W[7,1]$. Now, consider the least significant bit of X. This is the XOR of $W[7]$ and $W[8]$. We already know $W[7]$, so it is possible to compute $W[8]$. Once $W[8]$ is known, we can compute $W[9]$ by considering the second least significant bit of X. Continuing this way, we can get $W[14,7]$.

Now consider the quantity $(D+W) \oplus (W \ggg 1)$ for bit positions 7 to 13. If the possible carry bit into the addition $D+W$ at bit position 7 can be guessed, then $W[21,15]$ can be determined. Extending this reasoning further, we need to guess 7 carry bits and the initial 8 bits of W to completely determine W. If the obtained value of W satisifies $-W + \sigma_0(W) = D$, then we have the correct solution. The entire algorithm is shown in Figure 4.

In the algorithm, we use a function GTD, which takes low order $7i$ bits of W as input and produces low order $7i+7$ bits of W. This function is described at the end of the figure.

Fig. 3. Structure of W and $\sigma_0(W)$ for SHA-512.

W	w_{63}	w_{62}	w_{61}	w_{60}	w_{59}	w_{58}	w_{57}	w_{56}	w_{55}	w_{54}	w_{53}	w_{52}	w_{51}	w_{50}	w_{49}	w_{48}
$W \gg 7$	0	0	0	0	0	0	0	w_{63}	w_{62}	w_{61}	w_{60}	w_{59}	w_{58}	w_{57}	w_{56}	w_{55}
$W \ggg 1$	w_0	w_{63}	w_{62}	w_{61}	w_{60}	w_{59}	w_{58}	w_{57}	w_{56}	w_{55}	w_{54}	w_{53}	w_{52}	w_{51}	w_{50}	w_{49}
$W \ggg 8$	w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	w_{63}	w_{62}	w_{61}	w_{60}	w_{59}	w_{58}	w_{57}	w_{56}
W	w_{47}	w_{46}	w_{45}	w_{44}	w_{43}	w_{42}	w_{41}	w_{40}	w_{39}	w_{38}	w_{37}	w_{36}	w_{35}	w_{34}	w_{33}	w_{32}
$W \gg 7$	w_{54}	w_{53}	w_{52}	w_{51}	w_{50}	w_{49}	w_{48}	w_{47}	w_{46}	w_{45}	w_{44}	w_{43}	w_{42}	w_{41}	w_{40}	w_{39}
$W \ggg 1$	w_{48}	w_{47}	w_{46}	w_{45}	w_{44}	w_{43}	w_{42}	w_{41}	w_{40}	w_{39}	w_{38}	w_{37}	w_{36}	w_{35}	w_{34}	w_{33}
$W \ggg 8$	w_{55}	w_{54}	w_{53}	w_{52}	w_{51}	w_{50}	w_{49}	w_{48}	w_{47}	w_{46}	w_{45}	w_{44}	w_{43}	w_{42}	w_{41}	w_{40}
W	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}	w_{16}
$W \gg 7$	w_{38}	w_{37}	w_{36}	w_{35}	w_{34}	w_{33}	w_{32}	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}
$W \ggg 1$	w_{32}	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}
$W \ggg 8$	w_{39}	w_{38}	w_{37}	w_{36}	w_{35}	w_{34}	w_{33}	w_{32}	w_{31}	w_{30}	w_{29}	w_{28}	w_{27}	w_{26}	w_{25}	w_{24}
W	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0
$W \gg 7$	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7
$W \ggg 1$	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8	w_7	w_6	w_5	w_4	w_3	w_2	w_1
$W \ggg 8$	w_{23}	w_{22}	w_{21}	w_{20}	w_{19}	w_{18}	w_{17}	w_{16}	w_{15}	w_{14}	w_{13}	w_{12}	w_{11}	w_{10}	w_9	w_8

Fig. 4. A guess-then-determine algorithm for solving $D = -W + \sigma_0(W)$ for SHA-512.

1. Guess $W[7,0]$ and carry bits $c_1, c_2, c_3, c_4, c_5, c_6, c_7$.
2. Let $c_0 = 0$.
3. for($i = 0$; $i \leq 7$; i++)
4. $\quad W[7i+7, 0] = \mathrm{GTD}(W[7i, 7i-6], c_i)$;
5. If $-W + \sigma_0(W) = D$, then output W as one solution.

1. function GTD($W[7i, 7i-6], c_i$){
2. $X = (((D \gg (7i-7))\&(\texttt{7f})) + c_i + (W \gg (7i-7))\&(\texttt{7f})) \oplus ((W \gg (7i-6))\&(\texttt{7f}))$;
3. $\quad T_1 = (X\&\texttt{1}) \oplus ((W \gg (7i))\&\texttt{1})$;
4. $\quad T_2 = ((X \gg 1)\&\texttt{1}) \oplus T1$;
5. $\quad T_3 = ((X \gg 2)\&\texttt{1}) \oplus T2$;
6. $\quad T_4 = ((X \gg 3)\&\texttt{1}) \oplus T3$;
7. $\quad T_5 = ((X \gg 4)\&\texttt{1}) \oplus T4$;
8. $\quad T_6 = ((X \gg 5)\&\texttt{1}) \oplus T5$;
9. $\quad T_7 = ((X \gg 6)\&\texttt{1}) \oplus T6$;
10. temp $= T1 \oplus (T2 \ll 1) \oplus (T3 \ll 2) \oplus (T4 \ll 3) \oplus (T5 \ll 4) \oplus (T6 \ll 5) \oplus (T7 \ll 6)$;
11. $W[7i+7, 0] = W[7i, 7i-6] \oplus (temp \ll (7i+1))$;
12. Return $W[7i+7, 0]$.

This algorithm involves guessing $W[7,0]$ and bits $c_1, c_2, \ldots c_7$, which is a total of 15 bits. If the equation $D = -W + \sigma_0(W)$ does not have any solution, then none will be returned by this algorithm; on the other hand, if there is a solution or there are more than one solutions, then all solutions will be returned. A total of 2^{15} operations are required. The time for each operation is significantly less than the time for a single SHA-512 step and by our assessment it is about 2^{-4} times the time for a single SHA-512 step.

C Colliding Message Pairs

Colliding message pairs for 22-step SHA-512 and 22-step SHA-256 generated by the algorithm of Table 15 are provided in Tables 8 and 9 respectively. Examples of colliding message pairs for 23-step and 24-step SHA-256 are shown in Tables 10, 11 and 12.

Table 8. Colliding message pair for 22-step SHA-512 with standard IV. These messages have been generated using the algorithm of Table 15.

W_1	0-3	0000000000000000	0000000000000000	c2bc8e9a85e2eb5a	6d623c5d5a2a1442
	4-7	cd38e6dee1458de7	acb73305cddb1207	148f31a512bbade5	ecd66ba86d4ab7e9
	8-11	92aafb1e9cfa1fcb	533c19b80a7c8968	e3ce7a41b11b4d75	aef3823c2a004b20
	12-15	8d41a28b0d847692	7f214e01c4e96950	0000000000000000	0000000000000000
W_2	0-3	0000000000000000	0000000000000000	c2bc8e9a85e2eb5a	6d623c5d5a2a1442
	4-7	cd38e6dee1458de7	acb73305cddb1207	148f31a512bbade5	ecd66ba86d4ab7ea
	8-11	90668fd7ec6718ee	533c19b80a7c8968	dfce7a41b11b4d76	aef3823c2a004b20
	12-15	8d41a28b0d847692	7f214e01c4e96950	0000000000000000	ffffffffffffffff

Table 9. Colliding message pair for 22-step SHA-256 with standard IV. These messages have been generated using the algorithm of Table 15.

W_1	0-7	00000000	00000000	0be293bf	99c539c9	1c672194	99b6a58a	5bf1d0ae	0a9a18d3
	8-15	0c18cf1c	329b3e6e	dc4e7a43	ab33823f	8d41a28d	7f214e03	00000000	00000000
W_2	0-7	00000000	00000000	0be293bf	99c539c9	1c672194	99b6a58a	5bf1d0ae	0a9a18d4
	8-15	07d56809	329b3e6e	dc0e7a44	ab33823f	8d41a28d	7f214e03	00000000	ffffffff

Table 10. Colliding message pair for 23-step SHA-256 with standard IV. These messages utilize a single local collision starting at Step $i = 8$.

W_1	0-7	122060e3	000f813f	d92d3fc6	ea4a475f	fb0c6581	dc4558c4	d86428b4	6e2ca576
	8-15	c8d597bf	6372d4c2	ddbd721c	79d654c4	f0064002	a894b7b6	91b7628e	3224db20
W_2	0-7	122060e3	000f813f	d92d3fc6	ea4a475f	fb0c6581	dc4558c4	d86428b4	6e2ca576
	8-15	c8d597c0	6372d4c1	ddbd721c	78d6b4c5	f0064002	a894b7b6	91b7628e	3224db20

Table 11. Colliding message pair for 23-step SHA-256 with standard IV. These messages utilize a single local collision starting at Step $i = 9$.

W_1	0-7	c201bef2	14cc32c9	3b80da44	d8212037	8987161d	a790cb4a	53b8d726	89e9a288
	8-15	3edd76e0	05f41ddc	9ebc0fc3	e099698a	2eaec58f	e7060b78	95d7030d	6bf777c0
W_2	0-7	c201bef2	14cc32c9	3b80da44	d8212037	8987161d	a790cb4a	53b8d726	89e9a288
	8-15	3edd76e0	05f41ddd	9ebc0fc2	e099c98a	2daf2590	e7060b78	95d7030d	6bf777c0

Table 12. Colliding message pair for 24-step SHA-256 with standard IV. These messages utilize a single local collision starting at Step $i = 10$.

W_1	0-7	657adf63	06c066d7	90f0b709	95a3e1d1	c3017f24	fad6c2bf	dff43685	6abff0da
	8-15	e6cfc63f	de8fb4c1	c20ca05b	f74815cc	c2e789d9	208e7105	cc08b6cf	70171840
W_2	0-7	657adf63	06c066d7	90f0b709	95a3e1d1	c3017f24	fad6c2bf	dff43685	6abff0da
	8-15	e6cfc63f	de8fb4c1	c20ca05c	f74815cb	c2e7e9d9	1f8ed106	cc08b6cf	70171840

Table 13. Colliding message pair for 23-step SHA-512 with standard IV. These messages utilize a single local collision starting at Step $i = 8$.

W_1	0-3	b9fa6fc4729ca55c	8718310e1b3590e1	1d3d530cb075b721	99166b30ecbdd705
	4-7	27ed55b66c090b62	754b2163ff6feec5	6685f40fd8ab08f8	590c1c0522f6fdfd
	8-11	b947bb4013b688c1	d9d72ca8ab1cac04	69d0e120220d4edc	30a2e93aeef24e3f
	12-15	84e76299718478b9	f11ae711647763e5	d621d2687946e862	0ee57069123ecc8b
W_2	0-3	b9fa6fc4729ca55c	8718310e1b3590e1	1d3d530cb075b721	99166b30ecbdd705
	4-7	27ed55b66c090b62	754b2163ff6feec5	6685f40fd8ab08f8	590c1c0522f6fdfd
	8-11	b947bb4013b688c2	d9d72ca8ab1cac03	69d0e120220d4edc	30a3493aeef25076
	12-15	84e76299718478b9	f11ae711647763e5	d621d2687946e862	0ee57069123ecc8b

Table 14. Colliding message pair for 24-step SHA-512 with standard IV. These messages utilize a single local collision starting at Step $i = 10$.

W_1	0-3	dedb689cfc766965	c7b8e064ff720f7c	c136883560348c9c	3747df7d0cf47678
	4-7	855e17555cfedc5f	88566babccaa63e9	5dda9777938b73cd	b17b00574a4e4216
	8-11	86f3ff48fd12ea19	cd15c6f8d6da38ce	5e2c6b7b0411e70b	36ed67e93a794e66
	12-15	1b65e96b02767821	04d0950089db6c68	5bc9b9673e38eff3	b05d879ad024d3fa
W_2	0-3	dedb689cfc766965	c7b8e064ff720f7c	c136883560348c9c	3747df7d0cf47678
	4-7	855e17555cfedc5f	88566babccaa63e9	5dda9777938b73cd	b17b00574a4e4216
	8-11	86f3ff48fd12ea19	cd15c6f8d6da38ce	5e2c6b7b0411e70c	36ed67e93a794e65
	12-15	1b66096b02767829	04d0f50089db6e9f	5bc9b9673e38eff3	b05d879ad024d3fa

Table 15. Deterministic algorithm to obtain message pairs leading to collisions for 22-step SHA-2.

external `W_to_set_register_A(Step` i`, desired_a, Current State` $\{a_{i-1}, b_{i-1}, \ldots h_{i-1}\}$`)` :
Returns the required message W_i to be used in step i so that a_i is set to the given value.

external `W_to_set_register_E(Step` i`, desired_e, Current State` $\{a_{i-1}, b_{i-1}, \ldots h_{i-1}\}$`)` :
Returns the required message W_i to be used in step i so that e_i is set to the given value.

First Message words:

1. Select W_0, W_1, W_{14} and W_{15} randomly.
2. Set `DELTA` $= \sigma_1(W_{15}) - \sigma_1(W_{15} - 1)$.
3. Run Steps 0 and 1 of hash evaluation to define $\{a_1, b_1, \ldots h_1\}$.
4. Choose $W_2 =$ `W_to_set_register_A(2, DELTA` $- 1 + f_{MAJ}(-1, -2,$ `DELTA` $- 3)$`, {`$a_1, b_1, \ldots h_1$`})`.
5. Run Step 2 of hash evaluation to define $\{a_2, b_2, \ldots h_2\}$.
6. Choose $W_3 =$ `W_to_set_register_A(3, DELTA` $- 3$`, {`$a_2, b_2, \ldots h_2$`})`.
7. Run Step 3 of hash evaluation to define $\{a_3, b_3, \ldots h_3\}$.
8. Choose $W_4 =$ `W_to_set_register_A(4,` -2`, {`$a_3, b_3, \ldots h_3$`})`.
9. Run Step 4 of hash evaluation to define $\{a_4, b_4, \ldots h_4\}$.
10. Choose $W_5 =$ `W_to_set_register_A(5,` -1`, {`$a_4, b_4, \ldots h_4$`})`.
11. Run Step 5 of hash evaluation to define $\{a_5, b_5, \ldots h_5\}$.
12. Choose $W_6 =$ `W_to_set_register_A(6,` -1`, {`$a_5, b_5, \ldots h_5$`})`.
13. Run Step 6 of hash evaluation to define $\{a_6, b_6, \ldots h_6\}$.
14. Choose $W_7 =$ `W_to_set_register_A(7,` -1`, {`$a_6, b_6, \ldots h_6$`})`.
15. Run Step 7 of hash evaluation to define $\{a_7, b_7, \ldots h_7\}$.
16. Choose $W_8 =$ `W_to_set_register_A(8,` 0`, {`$a_7, b_7, \ldots h_7$`})`.
17. Run Step 8 of hash evaluation to define $\{a_8, b_8, \ldots h_8\}$.
18. Choose $W_9 =$ `W_to_set_register_A(9,` 0`, {`$a_8, b_8, \ldots h_8$`})`.
19. Run Step 9 of hash evaluation to define $\{a_9, b_9, \ldots h_9\}$.
20. Choose $W_{10} =$ `W_to_set_register_E(10,` -1`, {`$a_9, b_9, \ldots h_9$`})`.
21. Run Step 10 of hash evaluation to define $\{a_{10}, b_{10}, \ldots h_{10}\}$.
22. Choose $W_{11} =$ `W_to_set_register_E(11,` -1`, {`$a_{10}, b_{10}, \ldots h_{10}$`})`.
23. Run Step 11 of hash evaluation to define $\{a_{11}, b_{11}, \ldots h_{11}\}$.
24. Choose $W_{12} =$ `W_to_set_register_E(12,` -1`, {`$a_{11}, b_{11}, \ldots h_{11}$`})`.
25. Run Step 12 of hash evaluation to define $\{a_{12}, b_{12}, \ldots h_{12}\}$.
26. Choose $W_{13} =$ `W_to_set_register_E(13,` -1`, {`$a_{12}, b_{12}, \ldots h_{12}$`})`.

Second message words:

27. Define $\delta W_i = 0$ for $i \in \{0, 1, 2, 3, 4, 5, 6, 9, 11, 12, 13, 14\}$.
28. Define $\delta W_7 = 1$ and $\delta W_{15} = -1$.
29. Define $\delta W_8 = -1 - f_{IF}(e_7 + 1, f_7, g_7) + f_{IF}(e_7, f_7, g_7) - \Sigma_1(e_7 + 1) + \Sigma_1(e_7)$. (Refer (7))
30. Define $\delta W_{10} = -f_{IF}(e_9 - 1, f_9 - 1, g_9 + 1) + f_{IF}(e_9, f_9, g_9) - \Sigma_1(e_9 - 1) + \Sigma_1(e_9)$. (Refer (8))
31. Compute $W'_i = W_i + \delta W_i$ for $0 \leq i \leq 15$.