New Records in Collision Attacks on SHA-2

Yingxin Li[1], Fukang Liu[2], and Gaoli Wang[1](✉)

[1] Shanghai Key Laboratory of Trustworthy Computing, Software Engineering
Institute, East China Normal University, Shanghai, China
liyx1140@163.com,glwang@sei.ecnu.edu.cn
[2] Tokyo Institute of Technology, Tokyo, Japan
liu.f.ad@m.titech.ac.jp

Abstract. The SHA-2 family including SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and SHA512/256 is a U.S. federal standard published by NIST. Especially, there is no doubt that SHA-256 is one of the most important hash functions used in real-world applications. Due to its complex design compared with SHA-1, there is almost no progress in collision attacks on SHA-2 after ASIACRYPT 2015. In this work, we retake this challenge and aim to significantly improve collision attacks on the SHA-2 family. First, we observe from many existing attacks on SHA-2 that the current advanced tool to search for SHA-2 characteristics has reached the bottleneck. Specifically, longer differential characteristics could not be found, and this causes that the collision attack could not reach more steps. To address this issue, we adopt Liu et al.'s MILP-based method and implement it with SAT/SMT for SHA-2, where we also add more techniques to detect contradictions in SHA-2 characteristics. This answers an open problem left in Liu et al.'s paper to apply the technique to SHA-2. With this SAT/SMT-based tool, we search for SHA-2 characteristics by controlling its sparsity in a dedicated way. As a result, we successfully find the first practical semi-free-start (SFS) colliding message pair for 39-step SHA-256, improving the best 38-step SFS collision attack published at EUROCRYPT 2013. In addition, we also report the first practical free-start (FS) collision attack on 40-step SHA-224, while the previously best theoretic 40-step attack has time complexity 2^{110}. Moreover, for the first time, we can mount practical and theoretic collision attacks on 28-step and 31-step SHA-512, respectively, which improve the best collision attack only reaching 27 steps of SHA-512 at ASIACRYPT 2015. In a word, with new techniques to find SHA-2 characteristics, we have made some notable progress in the analysis of SHA-2 after the major achievements made at EUROCRYPT 2013 and ASIACRYPT 2015.

Keywords: practical collision attack · SHA-2 · SAT/SMT

1 Introduction

Before the devastating attacks in 2005 [37,38,39,40] on the MD-SHA hash family, there was a trend to design fast hash functions with a similar structure to MD4, including MD5, SHA-0, SHA-1, SHA-2, RIPEMD-128 and RIPEMD-160, just to name a few. After 2005, we have witnessed efficient collision attacks on full MD4 [37], MD5 [39], SHA-0 [2,40], and SHA-1 [15,16,35,38] as well as the SFS collision attack on full RIPEMD-128 [14]. In spite of these successful attacks on the MD-SHA hash family, SHA-2 survived this game, mainly due to its more conservative and complex design. Since SHA-2 has been used worldwide, studying its collision and preimage resistances is always of practical interest, though it is also challenging.

Preimage attacks on SHA-2. In the past few years, there have been many results for the preimage attacks on SHA-256 and SHA-512. The first preimage attack on SHA-256 and SHA-512 [11] based on the meet-in-the-middle (MITM) technique reached 24 steps with a complexity of about 2^{240} and 2^{480}, respectively. These preimage attacks were significantly improved at ASIACRYPT 2009 [1], which were improved to 43-step SHA-256 and 46-step SHA-512, respectively. Then, at ASIACRYPT 2010, Guo et al. [9] presented advanced MITM preimage attacks on 42-step SHA-256 and SHA-512, respectively. At FSE 2012, the biclique technique was applied to find preimages of SHA-2 [12], where preimage attacks on 45-step SHA-256 and 50-step SHA-512 with time complexity of $2^{255.5}$ and $2^{511.5}$ were achieved, respectively. It should be noted that the authors in [12] also presented pseudo-preimage attacks on 52-step SHA-256 and 57-step SHA-512 with a complexity of 2^{255} and 2^{511}, respectively. However, all these preimage attacks are far from practical.

Distinguishing attacks on the compression function of SHA-2. Compared with preimage and collision attacks, distinguishing attacks are less meaningful for a hash function, though they can help better understand its security. At the rump session of EUROCRYPT 2008 [42], the non-randomness of 39-step SHA-256 was presented, and a practical example for 33 steps was given by Yu and Wang. In [10], free-start (FS) near-collisions for up to 31 steps of SHA-256 were presented. Then, Lamberger and Mendel gave a second-order differential attack on 46 steps of SHA-256 with a practical complexity in [13]. Later, this attack was extended to 47 steps of SHA-256 with a practical complexity at ASIACRYPT 2011 [3]. At INSCRYPT 2014 [41], Yu and Bai further utilized the attack strategy in [3] to mount a practical distinguishing attack on 48 steps of SHA-512.

Collision attacks on SHA-2. The first practical collision attack on SHA-256 [29] was presented at FSE 2006, only reaching 18 steps. At FSE 2008, Nikolic and Biryukov [32] improved this practical attack to 21 steps, and they also gave a SFS collision attack on 23 steps of SHA-256. This attack was later further extended to 24 steps of SHA-256 and SHA-512 in [10,33]. Then, at ASIACRYPT 2011, the first major improvement was achieved, where the advanced guess-and-determine (GnD) technique to search for SHA-2 characteristics was invented [25], and the SFS collision for 32-step SHA-256 and the collision for 27-step SHA-256 were presented, respectively. After this work at ASIACRYPT 2011, this advanced automatic tool has been gradually improved in 3 papers published at EUROCRYPT 2013 [27], FSE 2014 [8] and ASIACRYPT 2015 [6]. In addition, much more complex message differences are used to mount (FS/SFS) collision attacks on SHA-2 in these 3 papers. A summary of these collision attacks is shown in Table 1.

Automatic tools to search for SHA-2 characteristics. Although major achievements have been made in collision attacks on SHA-2 in [6,8,25,27], the corresponding advanced automatic tool to find SHA-2 characteristics is not open-source. Due to the complex design of SHA-2, this significantly increased the difficulty to follow these works without this tool, let alone to improve this tool. Although Stevens open sourced his dedicated tools [34,35,36] to find MD5 and SHA-1 characteristics, they could not be applied to SHA-2 as SHA-2 is too complex, and contradictions easily occur in its differential characteristics [25]. Recently, to make finding collision-generating signed differential characteristics easier, Liu et al. invented a novel MILP-based method [23] and it works quite well for RIPEMD-160. As can be observed in [23], two main techniques are how to describe signed difference transitions through each component of the step function and how to automatically detect contradictions in an efficient way. At the end of [23], the authors left an interesting problem whether it is possible to apply this technique to SHA-2 because it is required for the model to detect more contradictions in SHA-2 characteristics.

Our contributions. We briefly summarize our contributions as follows:

1. We demonstrate for the first time that the technique developed in [23] can be applied to SHA-2, and this obviously gives a positive answer to the question left in [23]. Specifically, we develop a SAT/SMT-based tool to efficiently search for valid SHA-2 differential characteristics based on the technique to search for signed differential characteristics in [23] and the technique to automatically verify the correctness of a differential characteristic in [20].
2. We shed new insight into the (free-start/semi-free-start) collision attacks on SHA-2. For the first time, we are able to propose:
   - the first practical SFS colliding message pair for 39-step SHA-256, breaking the record of 38 steps kept by Mendel et al. at EUROCRYPT 2013 [27] after 10 years;
   - the first practical free-start colliding message pair for 40-step SHA-224, improving the previously best theoretic 40-step attack with time complexity 2^{110} published at FSE 2012 [17];
   - the first practical colliding message pair for 28-step SHA-512, updating the previously best record given at ASIACRYPT 2015 [6] by 1 step.
   - the first collision attack on 31-step SHA-512 with time complexity $2^{115.6}$, improving the previously best one published at ASIACRYPT 2015 [6] by 4 steps.

In addition to these notable progress, we also improved the best collision attack on 31-step SHA-256 published at EUROCRYPT 2013 [27], reducing the time complexity from $2^{65.5}$ to $2^{49.8}$. Our results are summarized in Table 1. Especially, we note that there is gap between the previous (SFS) collision attacks on SHA-256 and SHA-512. Specially, due to the similarity between SHA-256 and SHA-512, a (SFS) collision attack on r steps of SHA-256 should have been applicable to r steps of SHA-512, and vice versa. However, this is not the case in previous attacks, as shown in Table 1. We believe this is caused by the infeasibility to find the corresponding valid SHA-2 characteristics with the current GnD technique. Based on our new technique, we have made the (SFS) collision attacks on SHA-256 and SHA-512 reach the same number of steps.

Moreover, based on our results for SHA-2, it indicates that the SAT/SMT-based method performs much better than the dedicated but non-open-source ones developed in [6,8,25,27]. This also contradicts the claims made in [8] that the performance of SAT-based method for SHA-2 is bad. Note that our SAT/SMT-based method is completely different from the one used in [8], which simply uses a model to describe two parallel instances of the value transitions as in [31].

Table 1. Summary of collision attacks on SHA-2, where FS collision* denotes the free-start collision without considering padding, and SFS collision denotes the semi-free-start collision.

State size	Hash size	Attack type	Steps	Time	Memory	References	Year
256	All	collision	28	*practical*	\	[27]	2013
			31	$2^{65.5}$	2^{34}	[27]	2013
			31	$2^{49.8}$	2^{48}	**Sect. 4.2**	**2023**
		SFS collision	38	*practical*	\	[27]	2013
			39	***practical***	\	**Sect. 4.1**	**2023**
	256	FS collision	52	$2^{127.5}$	\	[17]	2012
	224	FS collision*	39	*practical*	\	[6]	2015
		FS collision	40	2^{110}	\	[17]	2012
		FS collision*	**40**	***practical***	\	**Sect. 4.5**	**2023**
512	All	collision	27	*practical*	\	[6]	2015
			28	***practical***	\	**Sect. 4.4**	**2023**
			31	$2^{115.6}$	$2^{77.3}$	**Sect. 4.3**	**2023**
		SFS collision	38	*practical*	\	[8]	2014
			39	*practical*	\	[6]	2015
	384	FS collision	40	2^{183}	\	[17]	2012
		FS collision*	41	*practical*	\	[6]	2015
	256	FS collision*	43	*practical*	\	[6]	2015
	224	FS collision*	44	*practical*	\	[6]	2015

The source code to search for the differential characteristics and verify the (SFS/FS) collisions for SHA-256 and SHA-512 is available at https://github.com/Peace9911/sha_2_attack.git

Outline. This paper is organized as follows. The notations and some preliminary works of this paper are introduced in Section 2. A high-level overview of how to implement the MILP-based method with an SAT/SMT-based method and how to overcome more contradictions in the differential characteristics of SHA-2 in is given Section 3. Then, we show how to find the differential characteristics to mount the (SFS/FS) collisions for SHA-2 in Section 4. Finally, we conclude this paper in Section 5.

2 Preliminaries

2.1 Notations

For a better understanding of this paper, we introduce the following notations.

1. $\boxplus$ and $\boxminus$ represent modulo addition and modulo subtraction on 32/64 bits, respectively.
2. $\gg$, $\ggg$, $\oplus$, $\neg$, $\vee$ and $\wedge$ represent *shift right*, *rotate right*, *exclusive or*, *not*, *or*, and *and*, respectively.
3. $x[i]$ denotes the i-th bit of x and $x[0]$ is the least significant bit.
4. δx denotes the modular difference, i.e., $\delta x = x' \boxminus x$.
5. Δx denotes the signed difference between x' and x. We use the same notation as in [21,23], i.e.,

$$\Delta x[i] = \begin{cases} \texttt{n} & (x[i] = 0, x'[i] = 1) \\ \texttt{u} & (x[i] = 1, x'[i] = 0) \\ \texttt{=} & (x[i] = x'[i]) \\ \texttt{0} & (x[i] = x'[i] = 0) \\ \texttt{1} & (x[i] = x'[i] = 1) \end{cases} \tag{1}$$

6. $M = (m_0, m_1, \ldots, m_{15})$ and $M' = (m'_0, m'_1, \ldots, m'_{15})$ represent two message blocks.

Definition 1. *[23] The signed difference Δx is said to be an expansion of the modular difference δx only when Δx corresponds to the modular difference δx.*

Definition 2. *[23] The hamming weight of the signed difference Δx is denoted by $\boldsymbol{H}(\Delta x)$ and $\boldsymbol{H}(\Delta x)$ is the number of indices i such that $\Delta x[i] \in \{\texttt{n}, \texttt{u}\}$.*

For example, let

$$\begin{aligned} \Delta x_0 &= [\texttt{==== nu== ==== ==== ==== ==== ==== ====}], \\ \Delta x_1 &= [\texttt{==== =n== ==== ==== ==== ==== ==== ====}]. \end{aligned}$$

Then, both Δx_0 and Δx_1 are the expansions of $\delta x = 2^{26}$. Moreover, we have $\mathbf{H}(\Delta x_0) = 2$ and $\mathbf{H}(\Delta x_1) = 1$. As each signed difference corresponds to a unique modular difference, for convenience, when computing $\delta x \boxplus \delta y$ for a given $(\Delta x, \Delta y)$, we also simply denote $\delta x \boxplus \delta y$ by $\Delta x \boxplus \Delta y$. For the above example, we have $\Delta x_0 \boxplus \Delta x_1 = 2^{27}$.

2.2 Description of SHA-2

The SHA-2 family is a series of hash functions standardized by NIST as part of the Secure Hash Standard (SHS) [7]. This family mainly consists of two versions, namely SHA-256 and SHA-512. Furthermore, NIST defines a general truncation procedure for SHA-256 and SHA-512, which includes SHA-224, SHA-512/224, SHA-512/256 and SHA-384. SHA-2 adopts the well-known Merkle-Damgård construction [5,30], and its compression functions employ the Davies-Meyer construction. As the two main versions of SHA-2, SHA-256 and SHA-512 have 32-bit and 64-bit state words, respectively. SHA-256 and SHA-512 utilize 512-bit message words and 1024-bit message words as input, with their chaining variables and final outputs being 256 bits and 512 bits, respectively.

The compression functions of SHA-256 and SHA-512 are computed through iterative updates to internal states. The number of steps, which is denoted by r, is 64 for SHA-256 and 80 for SHA-512. In the following, we provide a brief overview of their compression functions. They consist of two main parts: the message expansion and the state update transformation. A complete description of SHA-2 is given in [7].

Message Expansion. The 512-bit message block for SHA-256 and the 1024-bit message block for SHA-512 are divided into 16 message words of sizes 32 bits and 64 bits, respectively, which are denoted by $(m_1, \dots, m_{15})$. Then, the 16 message words are expanded to r expanded message words W_i, i.e., $W_0, W_1, \dots, W_{r-1}$:

$$W_i = \begin{cases} m_i & 0 \leq i \leq 15, \\ \sigma_1(W_{i-2}) \boxplus W_{i-7} \boxplus \sigma_0(W_{i-15}) \boxplus W_{i-16} & 16 \leq i \leq r-1. \end{cases}$$

The functions $\sigma_0(x)$ and $\sigma_1(x)$ in SHA-256 are given by

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

The functions $\sigma_0(x)$ and $\sigma_1(x)$ in SHA-512 are given by

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

State update transformation. We utilize the alternate description for the state update of SHA-256 and SHA-512, as illustrated in Figure 1.

The state update transformation starts from a 256-bit (resp. 512-bit) chaining value $iv = (A_{-1}, \dots, A_{-4}, E_{-1}, \dots, E_{-4})$ for SHA-256 (resp. SHA-512), and updates it by applying the step function r times. In each step $i = 0, \dots, r-1$, one expanded message word W_i is used to compute the two state words E_i and A_i as follows, where K_i is a predefined constant and can be referred to [7].

$$\begin{aligned} E_i &= A_{i-4} \boxplus E_{i-4} \boxplus \Sigma_1(E_{i-1}) \boxplus \mathrm{IF}(E_{i-1}, E_{i-2}, E_{i-3}) \boxplus K_i \boxplus W_i, \\ A_i &= E_i \boxminus A_{i-4} \boxplus \Sigma_0(A_{i-1}) \boxplus \mathrm{MAJ}(A_{i-1}, A_{i-2}, A_{i-3}). \end{aligned}$$

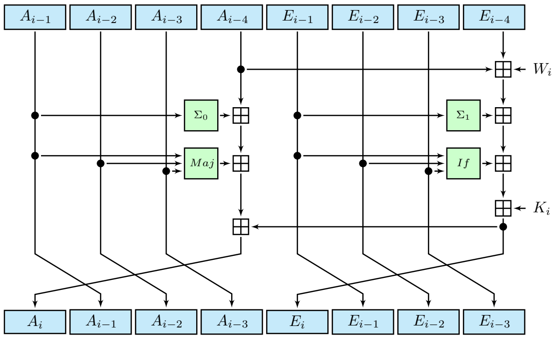

Fig. 1. The state update transformation of SHA-2.

Both SHA-256 and SHA-512 utilize the same Boolean functions IF and MAJ, as defined below:

$$\mathrm{IF}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus z,$$
$$\mathrm{MAJ}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z).$$

However, the linear functions Σ_0 and Σ_1 are different for SHA-256 and SHA-512. For SHA-256, they are defined below:

$$\Sigma_0(x) = (x \ggg 2) \oplus (x \ggg 13) \oplus (x \ggg 22),$$
$$\Sigma_1(x) = (x \ggg 6) \oplus (x \ggg 11) \oplus (x \ggg 25).$$

For SHA-512, they are defined below:

$$\Sigma_0(x) = (x \ggg 28) \oplus (x \ggg 34) \oplus (x \ggg 39),$$
$$\Sigma_1(x) = (x \ggg 14) \oplus (x \ggg 18) \oplus (x \ggg 41).$$

After the last step of the state update transformation, the previous chaining value is added to the output of the state update. The result of this feed-forward sum is the chaining value h:

$$h = (A_{63} \boxplus A_{-1}, \ldots, A_{60} \boxplus A_{-4}, E_{63} \boxplus E_{-1}, \ldots, E_{60} \boxplus E_{-4}).$$

On finding (FS/SFS) collisions. Denote the compression function of SHA-2 by $h_i = H(h_{i-1}, M_i)$. To find a collision with j message blocks, we need to find $(M_1, \ldots, M_j)$ and $(M'_1, \ldots, M'_j) \neq (M_1, \ldots, M_j)$ such that $h_j = h'_j$ where $h'_i = H(h'_{i-1}, M'_i)$ and $h_0 = h'_0$ is a *predefined constant*. In most cases, only $M_j \neq M'_j$ is required and we have $M_k = M'_k$ for $1 \leq k < j$. To find SFS collisions, we need to find $H(h, M) = H(h, M')$ where $M \neq M'$ and h can be an *arbitrary value*. To find FS collisions, we need to find $H(h, M) = H(h', M')$ where $M \neq M'$ and (h, h') can be *arbitrary values*.

2.3 Previous Methods to Search for Differential Characteristics

Almost all effective collision attacks on the MD-SHA hash family rely on Wang et al.'s techniques [37,38,39]. One of the most important steps is to find a collision-generating differential characteristic. For this purpose, there are three methods in the literature, as summarized below.

- **Hand-crafted method:** This remarkable work was first done by Wang et al. in their ground-breaking works on MD4 [37], MD5 [39], SHA-0 [40], and SHA-1 [38]. However, for complex designs like SHA-256 and RIPEMD-160, finding such differential characteristics for a large number of steps by hand is almost impossible, or at least considerably time-consuming.
- **Ad-hoc heuristic search tools:** De Cannière and Rechberger developed the first heuristic search tool for this problem based on the guess-and-determine (GnD) technique, and successfully applied it to SHA-1 [4]. Subsequently, this heuristic search tool were further developed and it has been applied to many hash functions like RIPEMD-128, RIPEMD-160, SHA-256, and SHA-512 [6,8,14,18,19,22,24,25,26,27,28]. However, the implementation of this GnD-based tool is not open-source. Although Stevens made his tools for MD5 and SHA-1 [34,35,36] open-source, it requires a significant amount of work to tweak them for SHA-2 because contradictions much more easily occur in the differential characteristics of SHA-2, and no existing tools for SHA-2 are based on this method.
- **Off-the-shelf solvers:** The method was first explored in [31] with SAT solvers after Wang et al.'s attacks and it was later also applied to SHA-1 in [35]. The main idea is to construct a model to describe two parallel instances of the value transitions. A new MILP-based method proposed by Liu et al. [23] is to model the pure signed difference transitions through each component of the round function, aided with some contradiction-detecting techniques. Especially, this technique [23] works quite well for RIPEMD-160.

3 SAT/SMT-based Tools for the MD-SHA Hash Family

The first SAT-based method to find collision-generating differential characteristics was proposed in 2006 [31], but the model is to simply describe two parallel instances of the value transitions. To efficiently capture the information of the signed difference propagation, the MILP-based method was proposed in [23]. Although the authors of [23] only target RIPEMD-160, since the MD-SHA hash functions share similar structures, the authors also mention that there are indeed much more applications beyond RIPEMD-160. Especially, whether it is applicable to SHA-2 is left as an interesting problem.

We answer this question in this paper. First, we show how to implement the MILP-based method [23] with an SAT/SMT-based method, and how to detect more contradictions in SHA-2 characteristics. Then, we demonstrate how to utilize our tools to find suitable differential characteristics to significantly improve the (SFS) collision attacks on SHA-2.

For the MILP-based method in [23], the constraints are already in Conjunctive Normal Form (CNF) due to the usage of the software Friday, which can output the minimized CNF for a given truth table with the Quine-McCluskey (QM) algorithm. However, they choose to further convert CNF into linear inequalities in order to use the solver Gurobi [23]. In this sense, we can not claim any novelty for how to re-implement the propagation of signed difference transitions with SAT/SMT. To make this paper self-contained, we briefly describe the idea to model the signed difference propagation with SAT/SMT. Note that when applying it to searching for valid SHA-2 characteristics, nontrivial additional techniques are required, as can be seen later in our detailed description of the search strategy.

For the MD-SHA hash family, it can be observed that in their round functions, there are three basic operations:

- modular addition;
- logic shift;
- Boolean functions.

Hence, we only describe how to describe the signed difference transitions through the modular addition and Boolean functions. For the logic shift, it does affect the model for RIPEMD-160 as shown in [23]. However, in the case of SHA-2, there is no such problem and it only affects the order of the variables. Hence, we simply omit it in this section.

Since we will target both SHA-256 and SHA-512, and their state sizes are 32 and 64 bits, respectively, to make the description of the model general, we treat the state size as n bits, i.e., the modular addition is within modulo 2^n.

3.1 SAT/SMT Models for the Signed Difference Transitions

Similar to [23], we use 2 binary variables (v, d) to describe the signed difference. Specifically, $(0, 0)$, $(0, 1)$ and $(1, 1)$ correspond to `[=]`, `[n]` and `[u]`, respectively, while we always exclude $(1, 0)$ as it carries the same information as $(0, 0)$. For the n-bit signed difference Δx, throughout this paper, the signed difference at the i-th $(0 \leq i \leq n-1)$ bit is always represented by $(x_v[i], x_d[i])$. For example, if $n = 5$ and $\Delta x = $ [`=u==n`], we have

$$(x_v[0], x_d[0]) = (0, 0), (x_v[1], x_d[1]) = (1, 1), (x_v[2], x_d[2]) = (0, 0),$$
$$(x_v[3], x_d[3]) = (0, 0), (x_v[4], x_d[4]) = (0, 1).$$

Modelling the modular addition. As explained in [23], given the signed difference Δx and Δy, it is sufficient to pick only 1 signed difference Δz to describe the modular difference $\delta z = \delta x \boxplus \delta y$.

To achieved this, the intermediate variable Δc with $\Delta c[0] = $ [=] is introduced and the propagation rules for

$$(\Delta x[i], \Delta y[i], \Delta c[i]) \xrightarrow{Add} (\Delta z[i], \Delta c[i+1])$$

Table 2. The propagation rules for $(\varDelta x[i], \varDelta y[i], \varDelta c[i]) \xrightarrow{Add} (\varDelta z[i], \varDelta c[i+1])$ in [23]

[=== → ==],	[==n → n=],	[==u → u=],	[=n= → n=],
[=u= → u=],	[=nn → =n],	[=un → ==],	[=nu → ==],
[=uu → =u],	[n== → n=],	[u== → u=],	[n=n → =n],
[u=n → ==],	[n=u → ==],	[u=u → =u],	[nn= → =n],
[nun → n=],	[unn → n=],	[nnu → n=],	[uun → u=],
[unu → u=],	[nuu → u=],	[uuu → uu].	

are shown in Table 2, where $0 \leq i \leq n-1$.

With the above method to describe the signed difference, there are 27 possible values for

$$(x_v[i], x_d[i], y_v[i], y_d[i], c_v[i], c_d[i], z_v[i], z_d[i], c_v[i+1], c_d[i+1])$$

based on Table 2. With the software LogicFriday, we can obtain the corresponding CNF to describe that this tuple can only take these 27 possible values. For convenience, we denote the CNF by $\mathcal{C}_{\texttt{Add}}(i)$. In this way, the complete model for the modular addition can be described with $\mathcal{C}_{Add}(i)$ for $0 \leq i \leq n-1$ and $(c_v[0], c_d[0]) = (0, 0)$.

For convenience, we denote the model for the modular addition $\delta z = \delta x \boxplus \delta y$ by $\mathcal{C}_{Add}(\varDelta x, \varDelta y, \varDelta z, \varDelta c)$.

Modelling the expansions of the modular difference [23]. In the above model, the signed difference transition through the modular addition is deterministic. To obtain all possible signed differences corresponding to the same modular difference, the authors of [23] introduce a model to describe the expansions of the modular difference. Given one $\varDelta z$, the aim is to find all possible $\varDelta\xi$ such that $\delta\xi = \delta z$, i.e., $\varDelta\xi$ and $\varDelta z$ correspond to the same modular difference. To achieve this, as in [23], an intermediate variable $\varDelta c$ is introduced and there are two methods to model it, as shown in Table 3.

Table 3. Two methods to describe the propagation rules for the expansion of modular difference [23]

Method 1 $(\varDelta z[i], \varDelta c[i]) \xrightarrow{Exp} (\varDelta\xi[i], \varDelta c[i+1])$	[nn → =n], [uu → =u], [nu → ==], [un → ==], [n= → n=], [n= → un], [u= → u=], [u= → nu], [=n → n=], [=n → un], [=u → u=], [=u → nu], [== → ==].
Method 2 $(\varDelta\xi[i], \varDelta z[i], \varDelta c[i]) \xrightarrow{Exp} (\varDelta c[i+1])$	[=un → n], [=nn → =], [=uu → =], [=nu → u], [u=n → =], [n=n → n], [u=u → u], [n=u → =], [nu= → n], [nn= → =], [uu= → =], [un= → u], [=== → =].

Similarly, based on the above way to describe the signed difference and using the software LogicFriday, the corresponding CNF to describe the constraints on

$$(z_v[i], z_d[i], c_v[i], c_d[i], \xi_v[i], \xi_d[i], c_v[i+1], c_d[i+1])$$

for Method 1 can be obtained, which is denoted by $\mathcal{C}_{Exp}(i)$. The complete model for the expansion of the modular difference is thus $\mathcal{C}_{Exp}(i)$ for $0 \leq i \leq n-1$ and $(c_v[0], c_d[0]) = (0, 0)$ for Method 1.

In the same way, we can also obtain the corresponding CNF denoted by $\mathcal{C}'_{Exp}(i)$ to describe the constraints on

$$(\xi_v[i], \xi_d[i], z_v[i], z_d[i], c_v[i], c_d[i], c_v[i+1], c_d[i+1])$$

for Method 2. The complete model for the expansion of the modular difference is thus $\mathcal{C}'_{Exp}(i)$ for $0 \leq i \leq n-1$ and $(c_v[0], c_d[0]) = (0, 0)$ for Method 2.

For convenience, we denote the model for the expansions of the modular addition in Method 1 and Method 2 by $\mathcal{C}_{Exp}(\Delta z, \Delta\xi, \Delta c)$ and $\mathcal{C}'_{Exp}(\Delta z, \Delta\xi, \Delta c)$.

Modelling the vectorial Boolean functions $w = f(x, y, z)$ [23]. In SHA-2, there are some vectorial Boolean functions, i.e., f can be XOR, IF or MAJ where $XOR(x, y, z) = x \oplus y \oplus z$. Note that σ_0, σ_1, Σ_0 and Σ_1 in SHA-2 are basically the same as XOR. Generally speaking, we can have

$$w[i] = f_i(x[i], y[i], z[i])$$

where f_i is a Boolean function $\mathbb{F}_2^3 \mapsto \mathbb{F}_2$ and $0 \leq i \leq n-1$. As described in [23], there are two models for $(f_i)_{0 \leq i \leq n-1}$: (i) the fast filtering model; (ii) the full model.

For the fast filtering model, we first need to build a table to include all valid propagation rules for $(\Delta x[i], \Delta y[i], \Delta z[i], \Delta w[i])$ and then obtain the corresponding valid values for

$$(x_v[i], x_d[i], y_v[i], y_d[i], z_v[i], z_d[i], w_v[i], w_d[i]).$$

Finally, LogicFriday is used to obtain the corresponding CNF for the constraints on this tuple.

For the full model, we need to involve both the signed difference and bit values. Specifically, the first step is to list all possible propagation rules for

$$(\Delta x[i], \Delta y[i], \Delta z[i], \Delta w[i], x[i], y[i], z[i]),$$

where $(x[i], y[i], z[i])$ can make the signed difference transition

$$(\Delta x[i], \Delta y[i], \Delta z[i]) \xrightarrow{f_i} \Delta w[i]$$

hold with probability 1. Then, we can obtain all the possible valid values for

$$(x_v[i], x_d[i], y_v[i], y_d[i], z_v[i], z_d[i], w_v[i], w_d[i], x[i], y[i], z[i]).$$

Finally, with LogicFriday, we obtain the corresponding CNF to describe the constraints on this tuple.

For convenience, we denote the fast filtering model and full model for $w = f(x, y, z)$ by $\mathcal{C}^f_{fast}(\Delta x, \Delta y, \Delta z, \Delta w)$ and $\mathcal{C}^f_{full}(\Delta x, \Delta y, \Delta z, \Delta w, x, y, z)$, respectively.

3.2 SAT/SMT Models for the Value Transitions

In SHA-2, contradictions easily occur in the collision-generating differential characteristics. To avoid this, we use the technique proposed by Liu et al. at CRYPTO 2020 [20]: using one model for the differential characteristic and another model for the value transitions. In the above model for the differential characteristic, we have included the relations between the value and the differential characteristic if using the full model for the Boolean functions. Specially, if the full model is applied to step i, the conditions on the internal states at step $i-1$, $i-2$ and $i-3$ to ensure the difference transitions have been added. Then, we can further build a model to optionally describe how to compute the internal state $i-1$ or $i-2$ or $i-3$ in order to test whether these conditions can hold, which is the model for the value transitions. It is easy to build the model for the value transitions as we only need to model the modular addition and Boolean functions.

To compute $z = x \boxplus y$, we can simply introduce a variable c with $c[0] = 0$ to denote the carry. Then, we list all possible values for the tuple $(x[i], y[i], c[i], z[i])$ and get the corresponding CNF for the model addition. For convenience, we denote the model for the modular addition of the value by $\mathcal{C}^{Add}_{Val}(x, y, z, c)$.

To compute $w = f(x, y, z)$, we can simply list all possible valid values for the tuple $(x[i], y[i], z[i], w[i])$ and get the corresponding CNF. For convenience, the model for the vectorial Boolean function f is denoted by $\mathcal{C}^f_{Val}(x, y, z, c)$.

With the two basic models $\mathcal{C}^{Add}_{Val}$ and $\mathcal{C}^f_{val}$, we can simply build the model for the value transitions through the step function of SHA-2 by decomposing the step function with intermediate variables. For convenience, the models to compute E_i, A_i and W_i are denoted by $\mathcal{C}^E_{Val}(i)$, $\mathcal{C}^A_{Val}(i)$ and $\mathcal{C}^W_{Val}(i)$, respectively.

Remark 1. With the model for value transitions, we can also use it to search for conforming input pairs for some dense parts of the differential characteristic. Specially, after a differential characteristic is obtained, we first derive all the differential conditions. Then, to find the conforming input pairs for the dense part of the characteristic, we simply use the value transitions for this part and add the corresponding differential conditions on the internal states to the model. This will be frequently used in our attacks in order to search for conforming message pairs automatically. Indeed, it is not surprising that this method has been used in [20,31].

3.3 Models for SHA-2

we can give a high-level description of the model for the step function of SHA-2, as shown in Algorithm 1. In this algorithm, we implicitly introduce many intermediate variables $(\Delta B_{i,j}, \Delta C_{i,j})$, where $\Delta B_{i,j}$ is used to decompose the step function and $\Delta C_{i,j}$ is used to denote the carry. Their concrete meanings should be clear from the context. In addition, we also provide several optional parameters (OP1,OP2,OP3,OP4,OP5,OP6,OP7,OP8) to control the search strategy to increase the flexibility of the model.

4 New (SFS/FS) Collision Attacks on SHA-2

In the (FS/SFS) collision attacks on SHA-2 [6,8,25,27] with the GnD tools, a crucial step is to first search for a relatively complex local collision in the message expansion, where nonzero message differences exist in the middle steps, and the differences will be cancelled in as many consecutive steps as possible in the forward and backward directions.

Basically, after determining the local collision in the message expansion, the number of attacked steps is also known. However, finding a valid attack further requires attackers to finish the following two tasks:

Task 1: searching for a corresponding differential characteristic in (A_i, E_i);
Task 2: finding the conforming message pair to ensure the validity of the differential characteristic since contradictions easily occur.

In some cases, even though we know there may exist a good local collision in the message expansion, it may be still infeasible to find a valid attack due to the difficulty of Task 1 or Task 2. For example, the SFS collision attack can reach 39 steps of SHA-512, but could not reach 39 steps of SHA-256. Moreover, the best collision attack on SHA-256 could reach 31 steps, while it is only 27 steps for SHA-512.

4.1 The First Practical SFS Collision for 39-Step SHA-256

We note that there is a practical SFS collision attack on 39-step SHA-512 published at ASIACRYPT 2015 [6]. However, the authors did not report any attacks on 39-step SHA-256, even though SHA-256 and SHA-512 share almost the same message expansion and state update function, i.e., only with different state sizes and different rotation numbers in Σ and σ. Specifically, the strategy to construct the local collision for 39-step SHA-512 should have been applicable to 39-step SHA-256, and this cannot be the bottleneck. We thus believe that the difficulty exists in either Task 1 or Task 2.

Hence, we aim to retake this challenge with the new SAT/SMT-based technique. First, we observe that in the differential characteristic for 39-step SHA-512 in [6], the local collision spans over 19 steps (steps $8-26$), and the nonzero message differences exist in 9 words $(W_8, \ldots, W_{12}, W_{16}, W_{17}, W_{24}, W_{26})$. In addition, in (W_{26}, W_{17}, A_{18}), there is only a one-bit difference, respectively.

In our new attack on 39-step SHA-256, we use the same strategy to construct the local collision, as shown in Figure 2(a). Different from the ad-hoc GnD techniques [6,8,27], it is efficient to use our SAT/SMT-based technique to find a sparse differential characteristic by minimizing the Hamming weight of the signed

Algorithm 1 High-level description of the model for the step function of SHA-2

```
procedure SHA2($i$,OP1,OP2,OP3,OP4,OP5,OP6,OP7,OP8)
    SHA2-E($i$,OP1,OP2,OP3)
    SHA2-A($i$,OP4,OP5,OP6)
    if $i \geq 16$ then
        SHA2-W($i$,OP7)
    if OP8==1 then
        $\mathcal{C}^{E}_{Val}(i)$, $\mathcal{C}^{A}_{Val}(i)$, $\mathcal{C}^{W}_{Val}(i)$
procedure SHA2-E($i$,OP1,OP2,OP3)
    $\mathcal{C}_{Add}(\Delta A_{i-4}, \Delta W_i, \Delta B_{i,0}, \Delta C_{i,0})$
    $\mathcal{C}_{Add}(\Delta E_{i-4}, \Delta B_{i,0}, \Delta B_{i,1}, \Delta C_{i,1})$
    $E^s_{i-1} = E_{i-1} \ggg s,$
    if OP1==1 then $\mathcal{C}^{XOR}_{fast}(\Delta E^2_{i-1}, \Delta E^{13}_{i-1}, \Delta E^{22}_{i-1}, \Delta B_{i,2})$
    else
        $\mathcal{C}^{XOR}_{full}(\Delta E^2_{i-1}, \Delta E^{13}_{i-1}, \Delta E^{22}_{i-1}, \Delta B_{i,2}, E^2_{i-1}, E^{13}_{i-1}, E^{22}_{i-1})$
    $\mathcal{C}_{Add}(\Delta B_{i,1}, \Delta B_{i,2}, \Delta B_{i,3}, \Delta C_{i,2})$
    if OP2==1 then $\mathcal{C}^{IF}_{fast}(\Delta E_{i-1}, \Delta E_{i-2}, \Delta E_{i-3}, \Delta B_{i,4})$
    else
        $\mathcal{C}^{IF}_{full}(\Delta E_{i-1}, \Delta E_{i-2}, \Delta E_{i-3}, \Delta B_{i,4}, E_{i-1}, E_{i-2}, E_{i-3})$
    $\mathcal{C}_{Add}(\Delta B_{i,3}, \Delta B_{i,4}, \Delta B_{i,5}, \Delta C_{i,3})$
    if OP3==1 then $\mathcal{C}_{Exp}(\Delta B_{i,5}, \Delta E_i, \Delta C_{i,4})$
    else
        $\mathcal{C}'_{Exp}(\Delta B_{i,5}, \Delta E_i, \Delta C_{i,4})$
procedure SHA2-A($i$,OP4,OP5,OP6)
    $A^s_{i-1} = A_{i-1} \ggg s,$
    if OP4==1 then $\mathcal{C}^{XOR}_{fast}(\Delta A^6_{i-1}, \Delta A^{11}_{i-1}, \Delta A^{25}_{i-1}, \Delta B_{i,6})$
    else
        $\mathcal{C}^{XOR}_{full}(\Delta A^6_{i-1}, \Delta A^{11}_{i-1}, \Delta A^{25}_{i-1}, \Delta B_{i,6}, A^6_{i-1}, A^{11}_{i-1}, A^{25}_{i-1})$
    if OP5==1 then $\mathcal{C}^{MAJ}_{fast}(\Delta A_{i-1}, \Delta A_{i-2}, \Delta A_{i-3}, \Delta B_{i,7})$
    else
        $\mathcal{C}^{MAJ}_{full}(\Delta A_{i-1}, \Delta A_{i-2}, \Delta A_{i-3}, \Delta B_{i,7}, A_{i-1}, A_{i-2}, A_{i-3})$
    $\mathcal{C}_{Add}(\Delta B_{i,6}, \Delta B_{i,7}, \Delta B_{i,8}, \Delta C_{i,5})$
    $\mathcal{C}_{Add}(\Delta A_{i-4}, \Delta B_{i,8}, \Delta B_{i,9}, \Delta C_{i,6})$
    $\mathcal{C}_{Add}(\Delta A_{i,9}, \Delta B_{i,10}, \Delta E_i, \Delta C_{i,7})$
    if OP6==1 then $\mathcal{C}_{Exp}(\Delta B_{i,10}, \Delta A_i, \Delta C_{i,8})$
    else
        $\mathcal{C}'_{Exp}(\Delta B_{i,10}, \Delta A_i, \Delta C_{i,8})$
procedure SHA2-W($i$,OP7)
    $W^s_{i-2} = W_{i-2} \ggg s, {W^s_{i-2}}' = W_{i-2} \gg s,$
    $\mathcal{C}^{XOR}_{full}(\Delta W^{17}_{i-2}, \Delta W^{19}_{i-2}, \Delta {W^{10}_{i-2}}', \Delta B_{i,10}, W^{17}_{i-2}, W^{19}_{i-2}, {W^{10}_{i-2}}')$
    $W^s_{i-15} = W_{i-15} \ggg s, {W^s_{i-15}}' = W_{i-15} \gg s,$
    $\mathcal{C}^{XOR}_{full}(\Delta W^{7}_{i-15}, \Delta W^{18}_{i-15}, \Delta {W^{3}_{i-15}}', \Delta B_{i,11}, W^{7}_{i-15}, W^{18}_{i-15}, {W^{3}_{i-15}}')$
    $\mathcal{C}_{Add}(\Delta B_{i,10}, \Delta W_{i-7}, \Delta B_{i,12}, \Delta C_{i,9})$
    $\mathcal{C}_{Add}(\Delta B_{i,11}, \Delta B_{i,12}, \Delta B_{i,13}, \Delta C_{i,10})$
    $\mathcal{C}_{Add}(\Delta B_{i,13}, \Delta W_{i-16}, \Delta B_{i,14}, \Delta C_{i,11})$
    if OP7==1 then $\mathcal{C}_{Exp}(\Delta B_{i,14}, \Delta W_i, \Delta C_{i,12})$
    else
        $\mathcal{C}'_{Exp}(\Delta B_{i,14}, \Delta W_i, \Delta C_{i,12})$
```

differences. This is crucial to improve the uncontrolled differential probability and to make the message modification more practical. Our general procedure to search for the differential characteristic for 39-step SHA-256 is summarized below:

Step 1: **Minimize the Hamming weight of $\varDelta W_i$.** Specifically, find the minimal value of $t_w = \sum_{i=0}^{38} \mathbf{H}(\varDelta W_i)$ such that the nonzero differences only exist in the 9 expanded message words $(W_8, \ldots, W_{12}, W_{16}, W_{17}, W_{24}, W_{26})$. Note that the concrete message differences are not specified at this step and the only goal is to find the minimal value t_w.

Step 2: **Minimize the Hamming weight of $\varDelta A_i$.** Specifically, under the conditions

$$\begin{aligned}\forall i \in [19, 38] :\ \ \delta A_i &= 0,\\ \forall i \in [23, 38] :\ \ \delta E_i &= 0,\\ \forall i \in [0, 38] \ \text{ and } \ i \notin \{8, \ldots, 12, 16, 17, 24, 26\} : \delta W_i &= 0,\\ \sum_{i=0}^{38} \mathbf{H}(\varDelta W_i) &= t_w,\end{aligned}$$

find the minimal value of $t_A = \sum_{i=0}^{38} \mathbf{H}(\varDelta A_i)$ such that there exists a solution of a 39-step collision-generating differential characteristic, i.e., there is a solution to $(\varDelta W_i, \varDelta A_i, \varDelta E_i)$ for $0 \leq i \leq 38$ to allow a 39-step attack. Still, we only aim at the minimal value t_A, and do not fix $(\varDelta W_i, \varDelta A_i, \varDelta E_i)$ according to the solution at this step.

Step 3: **Minimize the Hamming weight of $\varDelta E_i$.** In addition to the conditions at Step 2, we further add the condition

$$\sum_{i=0}^{38} \mathbf{H}(\varDelta A_i) = t_A.$$

Under these conditions, find and output the solution of $(\varDelta W_i, \varDelta A_i, \varDelta E_i)$ for $0 \leq i \leq 38$ that minimizes $\sum_{i=0}^{38} \mathbf{H}(\varDelta E_i)$.

Following the above procedure, we successfully found a corresponding 39-step differential characteristic, as shown in Table 4. By our procedure, this differential characteristic can be kept as sparse as possible and hence it is expected to be valid.

Remark 2. Our strategy to search for a concrete 39-step differential characteristic is different from the GnD technique in [6] because we first minimize the Hamming weight of $(\varDelta W_i, \varDelta A_i)$ and then search the solution under such constraints. However, there is no such a minimization procedure when searching for the differential characteristic in 39-step SHA-512 in [6]. Without this strategy, the differential characteristic may be dense and there is a high chance that it is invalid, which may somehow explain why the technique in [6] failed for 39-step SHA-256.

Message modification. As the differential characteristic is still relatively dense, we could not ensure that there must exist a conforming message pair. To verify this, we first extract all the constraints on $(A_i, E_i)_{-4\leq i\leq 22}$ and $(W_i)_{0\leq i\leq 38}$ for this differential characteristic. Then add these constraints to the SAT/SMT model for the value transitions of SHA-256, and solve the model to find a solution of these variables. We succeed in finding a practical SFS colliding message pair for 39-step SHA-256 in 120 seconds with 26 threads, as shown in Table 5.

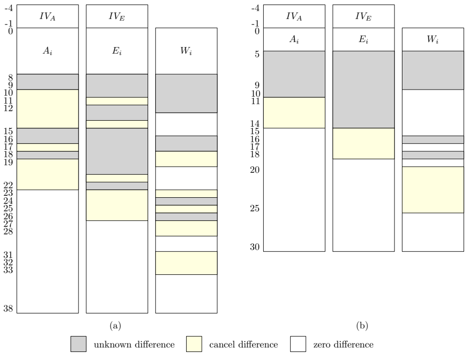

Fig. 2. (a) represent the shape of the 39-step differential SHA-256 and (b) represent the shape of the differential characteristic for 31-step SHA-256

4.2 Improved Collision Attacks on 31-Step SHA-256

The best existing collision attack on SHA-256 reaches 31 steps, which was published at EUROCRYPT 2013 [27]. The main idea is to use a two-block method to convert a SFS collision into a collision by utilizing the available degrees of freedom in the first few message words. To achieve this purpose, the first step is to find a suitable differential characteristic for 31-step SHA-256. In [27], this 31-step differential characteristic relies on a properly constructed local collision in the message expansion, which spans over 14 steps (steps 5−18). Specifically, the nonzero message differences only exist in 7 expanded message words

$$(W_5, W_6, W_7, W_8, W_9, W_{16}, W_{18}).$$

Moreover, there are no conditions on the first 5 expanded message words $(W_i)_{0\leq i\leq 4}$ and hence they can be freely chosen to efficiently convert a SFS collision into a collision. The shape of the 31-step differential characteristic is shown in Figure 2(b).

The method in [27] to convert SFS collisions into collisions is described below:

Step 1: Find 2^ℓ solutions of $(A_i)_{-3\leq i\leq 12}$, $(E_i)_{1\leq i\leq 12}$ and $(W_i)_{5\leq i\leq 12}$ that satisfy the differential conditions on steps 5−12. Store them in a table denoted by $\texttt{TAB}_1$.

Table 4. The differential characteristic for 39 steps of SHA-256

i	ΔA_i	ΔE_i	ΔW_i
-4	================================	================================	
-3	================================	================================	
-2	================================	================================	
-1	================================	================================	
0	================================	================================	================================
1	================================	================================	================================
2	================================	================================	================================
3	================================	================================	================================
4	================================	================================	================================
5	================================	================================	================================
6	================================	===0============================	================================
7	================================	===1=========11=====11======0===	================================
8	===u============================	unnn1=1110=0=0101==00011==11110=	===u============================
9	==============n=u====u======n===	010n0n0111010nu01001un011n10n=10	======n===u==========u==========
10	================================	0101u1n=1n0n010=u0=11nuu=1u00=n1	===n============================
11	================================	=100010000=0101=0===0010=10=1=0=	=======nn=======n===n===nn==uu=n
12	================================	=unn010000=1000011=00011==0=101=	=============u=======nn=========
13	================================	10110nuuuuuuuuu0u101un000010n111	================================
14	================================	=111=0000000000=0=1=001111111=1=	================================
15	============================n===	11001101101000000001nuuuuuuuu001	================================
16	=======u=u=======u==============	010100unu000001001u1000110unn=n1	======n===u==========u==========
17	================================	1100111u00nn=100110=u1u00unn000n	===n============================
18	===n============================	uuu1uuuu01000=110n000111101=0101	================================
19	================================	000u0n1000101=0un01=1100=u11n000	================================
20	================================	011100un0u001unnnn11000000001111	================================
21	================================	=110=111=0===000=1=======1==1===	================================
22	================================	=nuu==0110===00101=0110=====110=	================================
23	================================	=000============================	================================
24	================================	=111============================	=======n=n=======n==============
25	================================	================================	================================
26	================================	================================	===u============================
27	================================	================================	================================
28	================================	================================	================================
29	================================	================================	================================
30	================================	================================	================================
31	================================	================================	================================
32	================================	================================	================================
33	================================	================================	================================
34	================================	================================	================================
35	================================	================================	================================
36	================================	================================	================================
37	================================	================================	================================
38	================================	================================	================================

Table 5. The SFS colliding message pair for 39 steps of SHA-256

cv	02b19d5a 88e1df04 5ea3c7b7 f2f7d1a4 86cb1b1f c8ee51a5 1b4d0541 651b92e7
M	c61d6de7 755336e8 5e61d618 18036de6 a79f2f1d f2b44c7b 4c0ef36b a85d45cf f72b8c2f 0def947c a0eab159 8021370c 4b0d8011 7aad07f6 33cd6902 3bad5d64
M'	c61d6de7 755336e8 5e61d618 18036de6 a79f2f1d f2b44c7b 4c0ef36b a85d45cf e72b8c2f 0fcf907c b0eab159 81a1bfc1 4b098611 7aad07f6 33cd6902 3bad5d64
hash	431cadcd ce6893bb d6c9689a 334854e8 3baae1ab 038a195a ccf54a19 1c40606d

Step 2: Compute $2^{96-\ell}$ arbitrary first message blocks and get $2^{96-\ell}$ chaining inputs $(A_{-4}, \ldots, A_{-1})$ and $(E_{-4}, \ldots, E_{-1})$. Check $\texttt{TAB}_1$ and find a match in (A_{-3}, A_{-2}, A_{-1}). Then, $(W_i)_{0\leq i\leq 4}$ and E_0 are all determined for this match.

Step 3: At this step, $(W_i)_{0\leq i\leq 12}$ have been fixed. Use the degrees of freedom in (W_{13}, W_{14}, W_{15}) to fulfill the remaining uncontrolled conditions on $(E_{13}, E_{14}, E_{15}, W_{16}, W_{18})$. If it fails, go to Step 2.

Supposing Step 3 succeeds with probability $2^{-\gamma}$, the time complexity for this two-block method to find a collision is $2^{96-\ell+\gamma} + 2^{\ell} \cdot T_{\texttt{tool}}$, where $T_{\texttt{tool}}$ denotes the time to find a solution of $(A_i)_{-3\leq i\leq 12}$, $(E_i)_{1\leq i\leq 12}$ and $(W_i)_{5\leq i\leq 12}$ at Step 1. The memory complexity is 2^{ℓ}. In [27], $\ell \approx 34$, $\gamma \approx 3.5$ and $T_{\texttt{tool}}$ is negligible. Hence the time complexity is estimated as $2^{65.5}$ and the memory complexity is 2^{34}.

According to the above analysis, it is clear that ℓ and γ should be improved to get better attacks. Moreover, the best time-memory trade-off cannot be achieved with their 31-step differential characteristic [27]. Note that the maximal value of ℓ is dominated by the number of differential conditions on steps 5−12 and hence we can expect a relatively larger ℓ with a sparser differential characteristic. Therefore, we are interested whether it is possible to find a new sparser differential characteristic with our tool that can help achieve the optimal time-memory trade-off, i.e., with time and memory complexity close to $2^{96/2} = 2^{48}$. The overall searching procedure is stated as follows:

1. **Minimize the Hamming weight of ΔW_i.** Specifically, find the minimal value of $t_w = \sum_{i=0}^{30} \mathbf{H}(\Delta W_i)$ while keeping the minimal $\mathbf{H}(\Delta W_{16})$ and the minimal $\mathbf{H}(\Delta W_{18})$ such that the nonzero differences only exist in the 7 expanded message words $(W_5, W_6, W_7, W_8, W_9, W_{16}, W_{18})$. Note that the concrete message differences are not specified at this step.
2. **Minimize the Hamming weight of ΔA_i.** Specifically, under the conditions
$$\begin{aligned} \forall i \in [11, 30]:\ & \delta A_i = 0, \\ \forall i \in [15, 30]:\ & \delta E_i = 0, \\ \forall i \in [0, 30] \text{ and } i \notin \{5, \ldots, 9, 16, 18\}:\ & \delta W_i = 0, \\ & \sum_{i=0}^{30} \mathbf{H}(\Delta W_i) = t_w, \end{aligned}$$
find the minimal value of $t_A = \sum_{i=0}^{30} \mathbf{H}(\Delta A_i)$ such that there is a solution to $(\Delta W_i, \Delta A_i, \Delta E_i)$ for $0 \leq i \leq 30$ to allow a 31-step attack. Still, we only aim at the minimal value t_A, and do not fix $(\Delta W_i, \Delta A_i, \Delta E_i)$ according to the solution at this step.
3. **Minimize the Hamming weight of ΔE_i.** In addition to the conditions at Step 2, we further add the condition
$$\sum_{i=0}^{30} \mathbf{H}(\Delta A_i) = t_A.$$

Under these conditions, find and output the solution minimizing $\sum_{i=0}^{30} \mathbf{H}(\varDelta E_i)$ to allow a 31-step attack.

As already mentioned in our SAT/SMT models, to further detect the contradictions caused by the complex relationship between (A_i, E_i, W_i), we sometimes add the value transitions at certain steps to ensure its validity. In our model for the 31-step differential characteristic, this strategy is applied to $(A_i, E_i, W_i)_{7\leq i\leq 10}$. Without this strategy, we found that the obtained differential characteristic was indeed invalid[3]. Our new 31-step differential characteristic is shown in Table 6.

Estimating $\boldsymbol{\ell}$ and $\boldsymbol{\gamma}$. We use a dedicated method to find valid solutions of $(A_i)_{-3\leq i\leq 12}$, $(E_i)_{1\leq i\leq 12}$ and $(W_i)_{5\leq i\leq 12}$ such that ℓ can be better estimated. First, use the model for the value transitions to find a solution of $(A_i)_{1\leq i\leq 12}$, $(E_i)_{5\leq i\leq 12}$ and $(W_i)_{9\leq i\leq 12}$ that satisfy the differential conditions on steps $5-12$. For simplicity, this solution is called a starting point for 31-step SHA-256. Due to

$$A_i = E_i \boxminus A_{i-4} \boxplus \varSigma_0(A_{i-1}) \boxplus \mathrm{MAJ}(A_{i-1}, A_{i-2}, A_{i-3}), \tag{2}$$

$(A_{-3},\ A_{-2},\ A_{-1},\ A_0)$ will then depend on $(E_1,\ E_2,\ E_3,\ E_4)$ for this starting point. Moreover, according to

$$E_i = A_{i-4} \boxplus E_{i-4} \boxplus \varSigma_1(E_{i-1}) \boxplus \mathrm{IF}(E_{i-1}, E_{i-2}, E_{i-3}) \boxplus K_i \boxplus W_i, \tag{3}$$

$(E_1,\ E_2,\ E_3,\ E_4)$ will depend on $(W_5,\ W_6,\ W_7,\ W_8)$ for this starting point. By analyzing the conditions on $(W_5,\ W_6,\ W_7,\ W_8)$ to ensure the local collision in the message expansion, we find that there are in total 2^{14}, 2^{23}, 2^{27} and 2^{25} possible values of W_5, W_6, W_7 and W_8, respectively. Since there are no conditions on (E_1, E_2) or $(A_{-3},\ A_{-2},\ A_{-1},\ A_0)$ for this differential characteristic to hold, we only need to check how many (W_7, W_8) are left to ensure the conditions on (E_3, E_4) for this starting point. Experiments suggest that there are 2^{11} valid (W_7, W_8) left. Hence, based on this starting point, we can expect to generate $2^{14+23+11} = 2^{48}$ valid solutions of $(A_i)_{-3\leq i\leq 12}$, $(E_i)_{1\leq i\leq 12}$ and $(W_i)_{5\leq i\leq 12}$. For γ, since we do not have enough degrees of freedom in (W_{13}, W_{14}, W_{15}), we found that $\gamma \approx 1.3$ by 100 tests. If we can generate 2^{ℓ_1} starting points, then we have $2^{\ell} = 2^{\ell_1+48}$. Hence, the time complexity of the new collision attack on 31-step SHA-256 is estimated as

$$2^{96-48-\ell_1+1.3} + 2^{48+\ell_1} + 2^{\ell_1} \cdot T_{\texttt{model}},$$

where $T_{\texttt{model}} \approx 2^{31.7}$ denotes the time to generate a starting point and is always negligible. With $\ell_1 = 0$, i.e., only using one starting point, the time complexity is about $2^{49.8}$ and the memory complexity is 2^{48}. With this improved attack, we are much closer to a practical collision attack on 31-step SHA-256 and the bottleneck is the memory consumption. A possible practical implementation is to use less memory at the cost of increased time complexity.

[3] When searching for the differential characteristic for 39-step SHA-256, this strategy was not applied because we found that the obtained differential characteristic was valid.

Table 6. The differential characteristic for 31 steps of SHA-256

i	ΔA_i	ΔE_i	ΔW_i
-4	================================	================================	
-3	================================	================================	
-2	================================	================================	
-1	================================	================================	
0	================================	================================	================================
1	================================	================================	================================
2	================================	================================	================================
3	================================	==========================10====	================================
4	================================	============0===0=========01===0	================================
5	===================n=unnnnnnn=n=	000111010001111110nu=11111unnnu1	================nuuu=======0=uu=
6	========n======================u	101011=11==0n0==u11110==1110011n	==========u=====u===u======n===u
7	===u===n==n========n=========n=u	un0u1100n=01u11111001u1=n110u10n	=u=u=======n=====n=nu=n=====nun=
8	=============================n==	1u01un0u0=1=1=11n=0=u0=001001u0=	=u=nn==========u===u===u==1=====
9	================================	01100001110=0=010===00=11101u0=1	================u==========1=u==
10	================u============u==	=1n1uuuuu0100=1un0=10unnnnnnn010	================================
11	================================	=01u1010uu1==11100===1000001n=0=	================================
12	================================	==110001=11====1n====0011110n=0=	================================
13	================================	===0====01======1===============	================================
14	================================	================u===========0u==	================================
15	================================	================0============1==	================================
16	================================	================1============1==	=============unnnunnnnnnnnnnnn==
17	================================	================================	================================
18	================================	================================	==============1=n=0==========n==
19	================================	================================	================================
20	================================	================================	================================
21	================================	================================	================================
22	================================	================================	================================
23	================================	================================	================================
24	================================	================================	================================
25	================================	================================	================================
26	================================	================================	================================
27	================================	================================	================================
28	================================	================================	================================
29	================================	================================	================================
30	================================	================================	================================

4.3 The First Collision Attack on 31-Step SHA-512

While the best existing collision attack on SHA-256 reaches 31 steps, the best collision attack on SHA-512 could only reach up to 27 steps, which was reported at ASIACRYPT 2015 [6]. The authors also stated in [6] that they could not find better collision attacks on SHA-512 because they could not find a suitable differential characteristic with their tools. In this part, we show how to overcome this obstacle.

Our practical SFS collision attack on 39-step SHA-256 benefits much from the practical SFS collision attack on 39-step SHA-512 due to their similarity. Hence, we feel interested to know whether it is possible to find a suitable differential characteristic for 31-step SHA-512 based on the collision attack on 31-step SHA-256 [27] with our new tool.

Specifically, similar to the 31-step attack on SHA-256, the nonzero message differences are injected in

$$(W_5, W_6, W_7, W_8, W_9, W_{16}, W_{18}),$$

and the local collision in the message expansion spans over 14 steps (steps 5−28), as shown in Figure 2(b). Similar to the collision attack on 31-step SHA-256, we first find SFS collisions and then convert them into collisions with the two-block method. The general procedure to convert SFS collisions into collisions is essentially the same and we refer the readers to the above improved attack on 31-step SHA-256.

The most challenging step to achieve the collision attack on 31-step SHA-512 is how to find a valid differential characteristic. In what follows, we describe how to use our tool to solve this problem.

Step 1: Find a solution of $(\varDelta W_i)_{0\leq i\leq 30}$ with the minimal $\sum_{i=0}^{30}\mathbf{H}(\varDelta W_i)$, while keeping the minimal $\mathbf{H}(\varDelta W_{16})$ and the minimal $\mathbf{H}(\varDelta W_{18})$, which allows a local collision in the message expansion.

Step 2: With the fixed solution of $(\varDelta W_i)_{0\leq i\leq 30}$ obtained at Step 1, find a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 30}$, which follows the shape of the 31-step differential characteristic shown in Figure 2(b). Here, set a threshold to $\sum_{i=0}^{30}\mathbf{H}(\varDelta A_i)$. Specifically, choose an integer tr and add the constraint

$$\sum_{i=0}^{30}\mathbf{H}(\varDelta A_i) \leq tr$$

to the model. If the solver cannot output a solution in a reasonable time, e.g., 72 hours, increase tr until a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 30}$ is found. Keep the solution of $(\varDelta A_i)_{0\leq i\leq 30}$.

Step 3: With the fixed solution of $(\varDelta A_i, \varDelta W_i)_{0\leq i\leq 30}$, find a valid solution of $(\varDelta E_i)_{0\leq i\leq 30}$ with the minimal $\sum_{i=0}^{30}\mathbf{H}(\varDelta E_i)$, which allows a 31-step collision attack.

It is found that the obtained 31-step differential characteristic is invalid. Therefore, we propose to use the following method to correct this obtained solution.

Step 1: Set $(\Delta E_i)_{5\leq i\leq 7}$ as unknown variables. For the remaining $(\Delta E_i)_{0\leq i\leq 30}$ where $i \notin \{5,6,7\}$, keep them the same as those in the obtained solution. For $(\Delta A_i)_{0\leq i\leq 30}$ and $(\Delta W_i)_{0\leq i\leq 30}$, they are also kept the same as those in the obtained solution.

Step 2: Add the constraints describing the value transitions for $(A_i, E_i, W_i)_{7\leq i\leq 12}$ to the model.

In summary, we utilize the degrees of freedom in $(\Delta A_i, \Delta E_i)_{5\leq i\leq 7}$ and the model for value transitions to correct an invalid 31-step differential characteristic. In our search, the corresponding 31-step differential characteristic is shown in Table 7.

Complexity evaluation. As already mentioned, the only challenge to achieve the collision attack on 31-step SHA-512 is to find a suitable differential characteristic. Once it is found, the two-block method for 31-step SHA-256 can be directly applied. For consistency, we use the same notation, i.e., use (ℓ, γ, ℓ_1) to describe the time complexity and memory complexity as in the above collision attack on 31-step SHA-256. For our 31-step differential characteristic, there are in total 2^{36}, 2^{26}, 2^{25} and 2^{43} possible values for W_5, W_6, W_7 and W_8, respectively. For each starting point, i.e., the solution of $(A_i)_{1\leq i\leq 12}$, $(E_i)_{5\leq i\leq 12}$ and $(W_i)_{9\leq i\leq 12}$, we have experimentally found that there are on average $2^{15.3}$ possible (W_7, W_8) that can make the conditions on (E_3, E_4) hold. Therefore, for each starting point, we can generate $2^{36+26+15.3} = 2^{77.3}$ candidate solutions of $(A_i)_{-3\leq i\leq 12}$, $(E_i)_{1\leq i\leq 12}$ and $(W_i)_{5\leq i\leq 12}$. For 2^{ℓ_1} starting points, we thus can expect to generate $2^{\ell} = 2^{\ell_1+77.3}$ such many solutions. For γ, similarly, we found $\gamma \approx 0.9$ according to 100 experiments. Since the time complexity to generate a starting point is negligible, the whole time complexity is estimated as

$$2^{64\times 3-(\ell_1+77.3)+0.9} + 2^{\ell_1+77.3}$$

and the memory complexity is $2^{\ell_1+77.3}$. With $\ell_1 = 0$, i.e., only one starting point, the time and memory complexity are $2^{115.6}$ and $2^{77.3}$, respectively.

4.4 The Practical Collision Attack on 28-Step SHA-512

Similar to the 28-step attack on SHA-256 [27], the nonzero message differences are injected in

$$(W_8, W_9, W_{13}, W_{16}, W_{18}),$$

and the local collision in the message expansion spans over 11 steps (steps 8−18), resulting in a collision on 28-step SHA-512.

The most challenging step to achieve the collision attack on 28-step SHA-512 is how to find a valid differential characteristic. In what follows, we describe how to use our tool to solve this problem.

Step 1: Find a solution of $(\Delta W_i)_{0\leq i\leq 27}$ with the minimal $\sum_{i=0}^{27} \mathbf{H}(\Delta W_i)$ while keeping the minimal $\mathbf{H}(\Delta W_{16})$ and the minimal $\mathbf{H}(\Delta W_{18})$, which allows a local collision in the message expansion.

Table 7. The differential characteristic for 31-step SHA-512

i	ΔA_i	ΔE_i	ΔW_i
-4	==	==	
-3	==	==	
-2	==	==	
-1	==	==	
0	==	==	==
1	==	==	==
2	==	==	==
3	==	=11=00000====10=0=====000==101=0===1========0===1==1==011110====	==
4	==	=00011100=11101=1===10111==010=1===0===1=10=01=00==101101011=10=	==
5	=nunuuuuuu======u======u===unnnu===u=n======u=====nn===unnnn====	uuu1nuuuuu101unnn10001ununnnnn1n=1unnu10=010u011nu0u11unnnnn1111	==n======n=====un======u=====u=nunnnnu======u=====nn=======u====
6	========n=======u=================un=======u=====unnnn====nuunn=	00nnnnnnu0unu0u0n111unnn101u010u0011110u0un0nn0uu11uun10n0nn1nu1	n0u011=01u=====0n====10n====un=============n====uu===u=u=nuuuuu=
7	=n======nunnu==u=u=========u==uu=n=u========nnn====n=====u=u====	u1uu11u1n0nuu1000nnu10uu1un1uu0u110un111110nu1uu0n1010011nn=uu1u	=u=============n============u====u==unn==n====u=======n=0=1====u
8	===u	11110011001100u000100u1n00n0011==n=0000n=0nu0un0n1n00010n0111110	n=====nu=====u=====nu=====n======u=====u======u==========0======
9	========================n=======================================	11n1==111=010111011101unn01000u0=011u1u00=0110010101=1==10101101	============u============n============n========================n
10	============u===========nu============n======================nuu	==10==1===111=10u0==101=0n0==11==11010n11=1u=0110=0=0n==1101=nuu	==
11	==	=1u1======un0=n=nn===11001u==un====unnn0n==n=========u==u=10=100	==
12	==	=000======00n=1=11===10==u1==00===1010u11==1=========1==1==0=11u	==
13	==	==1=======110=1=01=======11==11====1111=1==0=========0==1======1	==
14	==	============u============n============n========================n	==
15	==	============0============0============0========================0	==
16	==	============1============1============1========================1	=======unnnnn====nuuuuuuuu=======nuuuuu==========11====nuuuuuuuu
17	==	==	==
18	==	==	============n============u============u========================u
19	==	==	==
20	==	==	==
21	==	==	==
22	==	==	==
23	==	==	==
24	==	==	==
25	==	==	==
26	==	==	==
27	==	==	==
28	==	==	==
29	==	==	==
30	==	==	==

Step 2: **Find the suitable $\varDelta E_i$.** With the fixed solution of $(\varDelta W_i)_{0\leq i\leq 27}$ obtained at Step 1, find a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 27}$.

To improve the efficiency of the message modification, we have tried three strategies for Step 2, as detailed below:

Strategy 1: First, with the fixed solution of $(\varDelta W_i)_{0\leq i\leq 27}$, find a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 27}$, and we minimize $\sum_{i=0}^{27}\mathbf{H}(\varDelta A_i)$.
Then, with the fixed solution of $(\varDelta W_i, \varDelta A_i)_{0\leq i\leq 27}$, find a valid solution of $(\varDelta E_i)_{0\leq i\leq 27}$ with the minimal $\sum_{i=0}^{27}\mathbf{H}(\varDelta E_i)$.
Strategy 2: With the fixed solution of $(\varDelta W_i)_{0\leq i\leq 27}$, find a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 27}$, and we minimize $\sum_{i=0}^{27}\mathbf{H}(\varDelta E_i)$.
Strategy 3: With the fixed solution of $(\varDelta W_i)_{0\leq i\leq 27}$, find a valid solution of $(\varDelta A_i, \varDelta E_i)_{0\leq i\leq 27}$, and we minimize $\sum_{i=11}^{27}\mathbf{H}(\varDelta E_i)$.

After testing, it is found that Strategy 3 is more suitable for message modifications. However, such a 28-step differential characteristic is invalid. Similar to the method to correct the SHA-512 31-step differential characteristic, we also use the same technique to correct this invalid 28-step differential characteristic.

Step 1: Set $(\varDelta E_i)_{8\leq i\leq 10}$ as unknown variables. For the remaining $(\varDelta E_i)_{0\leq i\leq 27}$ where $i \notin \{8, 9, 10\}$, keep them the same as those in the obtained solution. For $(\varDelta A_i)_{0\leq i\leq 27}$ and $(\varDelta W_i)_{0\leq i\leq 27}$, they are also kept the same as those in the obtained solution.
Step 2: Add the constraints describing the value transitions for $(A_i, E_i, W_i)_{10\leq i\leq 12}$ to the model.

With this method, we eventually found a valid 28-step differential characteristic, as shown in Table 8.

Message modification. We use a different message modification technique than in [27]. In our message modification technique, we first determine all expanded message words and state variables in steps $8-12$. Since the first 8 message words can be (almost) freely chosen, it is easy to connect the $(A_i, E_i)_{-4\leq i\leq -1}$ and $(A_i, E_i)_{8\leq i\leq 12}$ by using $(W_i)_{0\leq i\leq 7}$. Currently, $(A_i, E_i)_{-4\leq i\leq 12}$ and $(W_i)_{0\leq i\leq 12}$ has been determined. Then, the degree of freedom in message words $W_{13} - W_{15}$ can be used to fulfill the conditions on $E_{13} - E_{15}$ and (W_{16}, W_{18}). With this method, the cost to find the colliding message pair is almost negligible. The colliding message pair is shown in Table 9.

4.5 The First Practical FS Collision for 40-step SHA-224

In SHA-224, the last one output word $(E_{60} + E_{-4})$ was truncated. Therefore, similar to [6], we inject differences in E_{-4} to mount a FS collision attack. The best practical FS collision attack on SHA-224 was presented in [6] and it reaches

Table 8. The differential characteristic for 28-step SHA-512

i	ΔA_i	ΔE_i	ΔW_i
-4	==	==	
-3	==	==	
-2	==	==	
-1	==	==	
0	==	==	==
1	==	==	==
2	==	==	==
3	==	==	==
4	==	==	==
5	==	==	==
6	==	=100=10=1011101=100010011=1000101=0==0=011==110==1001100=======0	==
7	==	=001=11000100110010001010=111101001==1010111011000110000=0=0==01	==
8	====unnnuuuuu=n======n=nn=u=u=nuuunuu==u=u==nuu=nuu=u=n=u=====un	=unnnnn0nuuuunu1nuuuuununununnuuu0uuuunnun00nuu10nuunnuun101110u	======u=nuuuunu====nu=uunu=nn===nu=nuun=nn===nu==nu=u=n=u======u
9	====unnnn==nnnuuuuuuuuuuuuuuuuuuuuuu=====n====nuuu=nnuu======n==	1un01unn1nnunnuun0nuuuunu0unuu111un110ununununnnn0nn10nn1u1u01nn	1000001unnnnn0110011=nuuuu========nuuuu===========nuuuuuuuuuuuuu
10	======unnnnnn===========nu=======nuuuuunuuuuuuuuuuuuuuuuuuuuuuuu	=01000=10111u1=0111=0=101n101100010000111=01=111101101010010=nuu	==
11	==	==u0100unnn00n1n0n11010101u000u=000=1011n10u010000010u0un0111101	==
12	==	=100===0111n10=1=1===11==u1==10=======u=1=01====1===01=10====00u	==
13	==	==1====11110n1=101=======10===0=======0=1==1=====1===0=11======1	=========n============u============n=====================u==n===
14	==	===========0u============n========0===n========================n	==
15	==	============1============0============0========================0	==
16	==	============0============1============1========================1	=======unnnnn====nuuuuuuuu=======nuuuuu==========11====nuuuuuuuu
17	==	==	==
18	==	==	============n============u============u========================u
19	==	==	==
20	==	==	==
21	==	==	==
22	==	==	==
23	==	==	==
24	==	==	==
25	==	==	==
26	==	==	==
27	==	==	==

Table 9. The colliding message pair for 28 steps of SHA-512

M	1f736d69a0368ef6	7277e5081ad1c198	e953a3cdc4cbe577	bd05f6a203b2f75f
	dd18b3e39f563fca	cad0a5bb69049fcd	4d0dd2a06e2efdc0	86db19c26fc2e1cf
	0184949e92cdd314	82fb3c1420112000	e4930d9b8295ab26	5500d3a2f30a3402
	26f0aa8790cb1813	a9c09c5c5015bc0d	53892c5a64e94edb	8e60d500013a1932
M'	1f736d69a0368ef6	7277e5081ad1c198	e953a3cdc4cbe577	bd05f6a203b2f75f
	dd18b3e39f563fca	cad0a5bb69049fcd	4d0dd2a06e2efdc0	86db19c26fc2e1cf
	037a8f464c0bb995	83033bd41e111fff	e4930d9b8295ab26	5500d3a2f30a3402
	26f0aa8790cb1813	a9809e5c4015bc45	53892c5a64e94edb	8e60d500013a1932
hash	dceb3d88adf54bd2	966c4cb1ab0cf400	01e701fdf10ab603	796d6e5028a5e89a
	f29a7517b216c09f	46dbae73b1db8cce	8ea44d45041010ea	26a7a6b902f2632f

39 steps. With our tool, we could find a practical FS collision for 40-step SHA-224 for the first time. Specifically, we inject message differences at 10 expanded words

$$(W_0, W_9, W_{10}, W_{11}, W_{12}, W_{13}, W_{17}, W_{18}, W_{25}, W_{27}),$$

and then search for the corresponding 40-step differential characteristic. The searching strategy is almost the same as in our attack on 39-step SHA-256.

The 40-step differential characteristic and the conforming message pair are shown in Table 10 and Table 11, respectively.

5 Summary and Future Work

Although there was major progress on collision attacks on SHA-2 between 2011 and 2015, which essentially benefited from the development of the GnD technique to search for SHA-2 characteristics, no other progress has been made for nearly 8 years. One reason we believe is that the GnD technique has reached the bottleneck. In addition, the code for this GnD technique is not open source, which may further increase the difficulty to follow these works. Given the importance of SHA-2, there is no doubt that advancing the understanding of its collision resistance is always of practical interest.

By this work, we report for the first time that it is possible to overcome the obstacle to find SHA-2 characteristics with a SAT/SMT-based method, which is supported by several new improved attacks on the SHA-2 family. As can be observed, these new attacks highly depend on our SAT/SMT-based tool and how to use it in a dedicated way. Especially, we could find useful SHA-2 characteristics that could not be found with the GnD technique.

Through this work, we also expect that there could be more efforts to further improve this SAT/SMT-based method in the future, and that more and more researchers can easily perform analysis of SHA-2 with our tool.

Acknowledgement. We would like to thank the anonymous reviewers for their insightful comments. Yingxin Li and Gaoli Wang are supported by the National

Table 10. The differential characteristic for 40 steps of SHA-224

i	ΔA_i	ΔE_i	ΔW_i
-4	================================	===u============================	
-3	================================	================================	
-2	================================	================================	
-1	================================	================================	
0	================================	================================	===n============================
1	================================	================================	================================
2	================================	================================	================================
3	================================	================================	================================
4	================================	================================	================================
5	================================	================================	================================
6	================================	================================	================================
7	================================	0111============================	================================
8	================================	1000=========10======1===1==1===	================================
9	===u============================	unnn1=0=00=0=00=01=1=100=0110=1=	===u============================
10	==============n=u====u======n===	100n0n110111=nu00011un101n11n=00	======n===u==========u==========
11	================================	0101u0n=1n0n010=u0=10nun=1u01=n1	===n============================
12	================================	=10001000010001=0===0110=10=1=0=	=======nn=======n===n===nn==uu=u
13	================================	=unn00000001100011=00011==0=101=	=============u=======nn=========
14	================================	11100nuuuuuuuuu1u=01un000001n001	================================
15	================================	=111=0000000000=0=1=001111111=1=	================================
16	============================n===	11001101101000000101nuuuuuuuu001	================================
17	=======u=u=======u==============	010100unu000001001u1000110unn=n1	======n===u==========u==========
18	================================	1100111u00nn=100110=u1u00unn000n	===n============================
19	===n============================	uuu1uuuu01000=110n000111101=0101	================================
20	================================	000u0n1000101=0un01=1100=u11n000	================================
21	================================	011100un1u001unnnn11000000101111	================================
22	================================	=110=111=0===11101=======1==1===	================================
23	================================	=nuu==0110===00101=0110=====110=	================================
24	================================	=000============================	================================
25	================================	=111============================	=======n=n=======n==============
26	================================	================================	================================
27	================================	================================	===u============================
28	================================	================================	================================
29	================================	================================	================================
30	================================	================================	================================
31	================================	================================	================================
32	================================	================================	================================
33	================================	================================	================================
34	================================	================================	================================
35	================================	================================	================================
36	================================	================================	================================
37	================================	================================	================================
38	================================	================================	================================
39	================================	================================	================================

Table 11. The FS colliding message pair for 40 steps of SHA-224

CV	791c9c6b baa7f900 f7c53298 9073cbbd c90690c5 5591553c 43a5d984 af92402d
CV'	791c9c6b baa7f900 f7c53298 9073cbbd c90690c5 5591553c 43a5d984 bf92402d
M	f41d61b4 ce033ba2 dd1bc208 a268189b ee6bda2c 5ddbe94d 9675bbd3 32c1ba8a 7eba797d 88b06a8f 3bc3015c d36f38cc cfcb88e0 3c70f7f3 faa0c1fe 35c62535
M'	e41d61b4 ce033ba2 dd1bc208 a268189b ee6bda2c 5ddbe94d 9675bbd3 32c1ba8a 7eba797d 98b06a8f 39e3055c c36f38cc ce4b002d 3c74f1f3 faa0c1fe 35c62535
hash	9af50cac c165a72f b6f1c9f3 ef54bad9 af0cfb1f 57d357c9 c6462616

Key Research and Development Program of China (No. 2022YFB2701900), the National Natural Science Foundation of China (No. 62072181). Fukang Liu is supported by Grant-in-Aid for Research Activity Start-up (Grant No. 22K21282).

References

1. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for step-reduced SHA-2. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 5912, pp. 578–597. Springer (2009). https://doi.org/10.1007/978-3-642-10366-7_34
2. Biham, E., Chen, R., Joux, A., Carribault, P., Lemuet, C., Jalby, W.: Collisions of SHA-0 and reduced SHA-1. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 3494, pp. 36–57. Springer (2005), https://doi.org/10.1007/11426639_3
3. Biryukov, A., Lamberger, M., Mendel, F., Nikolic, I.: Second-order differential collisions for reduced SHA-256. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 7073, pp. 270–287. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_15
4. Cannière, C.D., Rechberger, C.: Finding SHA-1 characteristics: General results and applications. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 4284, pp. 1–20. Springer (2006), https://doi.org/10.1007/11935230_1
5. Damgård, I.: A design principle for hash functions. In: CRYPTO. Lecture Notes in Computer Science, vol. 435, pp. 416–427. Springer (1989), https://doi.org/10.1007/0-387-34805-0_39
6. Dobraunig, C., Eichlseder, M., Mendel, F.: Analysis of SHA-512/224 and SHA-512/256. In: ASIACRYPT(2). Lecture Notes in Computer Science, vol. 9453, pp. 612–630. Springer (2015), https://doi.org/10.1007/978-3-662-48800-3_25
7. Draft, F.: Public comments on the draft federal information processing standard (fips) draft fips 180-2, secure hash standard (shs)
8. Eichlseder, M., Mendel, F., Schläffer, M.: Branching heuristics in differential collision search with applications to SHA-512. In: FSE. Lecture Notes in Computer Science, vol. 8540, pp. 473–488. Springer (2014), https://doi.org/10.1007/978-3-662-46706-0_24
9. Guo, J., Ling, S., Rechberger, C., Wang, H.: Advanced meet-in-the-middle preimage attacks: First results on full tiger, and improved results on MD4 and SHA-2. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 6477, pp. 56–75. Springer (2010). https://doi.org/10.1007/978-3-642-17373-8_4
10. Indesteege, S., Mendel, F., Preneel, B., Rechberger, C.: Collisions and other non-random properties for step-reduced SHA-256. In: SAC. Lecture Notes in Computer Science, vol. 5381, pp. 276–293. Springer (2008). https://doi.org/10.1007/978-3-642-04159-4_18
11. Isobe, T., Shibutani, K.: Preimage attacks on reduced tiger and SHA-2. In: FSE. Lecture Notes in Computer Science, vol. 5665, pp. 139–155. Springer (2009). https://doi.org/10.1007/978-3-642-03317-9_9
12. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for preimages: Attacks on skein-512 and the SHA-2 family. In: FSE. Lecture Notes in Computer Science, vol. 7549, pp. 244–263. Springer (2012). https://doi.org/10.1007/978-3-642-34047-5_15
13. Lamberger, M., Mendel, F.: Higher-order differential attack on reduced SHA-256. IACR Cryptol. ePrint Arch. p. 37 (2011), http://eprint.iacr.org/2011/037

14. Landelle, F., Peyrin, T.: Cryptanalysis of full RIPEMD-128. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 7881, pp. 228–244. Springer (2013). https://doi.org/10.1007/978-3-642-38348-9_14
15. Leurent, G., Peyrin, T.: From collisions to chosen-prefix collisions application to full SHA-1. In: EUROCRYPT(3). Lecture Notes in Computer Science, vol. 11478, pp. 527–555. Springer (2019), https://doi.org/10.1007/978-3-030-17659-4_18
16. Leurent, G., Peyrin, T.: SHA-1 is a shambles: First chosen-prefix collision on SHA-1 and application to the PGP web of trust. In: USENIX. pp. 1839–1856. USENIX Association (2020), https://www.usenix.org/conference/usenixsecurity20/presentation/leurent
17. Li, J., Isobe, T., Shibutani, K.: Converting meet-in-the-middle preimage attack into pseudo collision attack: Application to SHA-2. In: FSE. Lecture Notes in Computer Science, vol. 7549, pp. 264–286. Springer (2012). https://doi.org/10.1007/978-3-642-34047-5_16
18. Liu, F., Dobraunig, C., Mendel, F., Isobe, T., Wang, G., Cao, Z.: Efficient collision attack frameworks for RIPEMD-160. In: CRYPTO(2). Lecture Notes in Computer Science, vol. 11693, pp. 117–149. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_5
19. Liu, F., Dobraunig, C., Mendel, F., Isobe, T., Wang, G., Cao, Z.: New semi-free-start collision attack framework for reduced RIPEMD-160. IACR Trans. Symmetric Cryptol. **2019**(3), 169–192 (2019). https://doi.org/10.13154/tosc.v2019.i3.169-192
20. Liu, F., Isobe, T., Meier, W.: Automatic verification of differential characteristics: Application to reduced gimli. In: CRYPTO. Lecture Notes in Computer Science, vol. 12172, pp. 219–248. Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_8
21. Liu, F., Meier, W., Sarkar, S., Wang, G., Ito, R., Isobe, T.: New cryptanalysis of ZUC-256 initialization using modular differences. IACR Trans. Symmetric Cryptol. **2022**(3), 152–190 (2022), https://doi.org/10.46586/tosc.v2022.i3.152-190
22. Liu, F., Mendel, F., Wang, G.: Collisions and semi-free-start collisions for round-reduced RIPEMD-160. In: ASIACRYPT(1). Lecture Notes in Computer Science, vol. 10624, pp. 158–186. Springer (2017), https://doi.org/10.1007/978-3-319-70694-8_6
23. Liu, F., Wang, G., Sarkar, S., Anand, R., Meier, W., Li, Y., Isobe, T.: Analysis of RIPEMD-160: new collision attacks and finding characteristics with MILP. In: EUROCRYPT(4). Lecture Notes in Computer Science, vol. 14007, pp. 189–219. Springer (2023). https://doi.org/10.1007/978-3-031-30634-1_7
24. Mendel, F., Nad, T., Scherz, S., Schläffer, M.: Differential attacks on reduced RIPEMD-160. In: ISC. Lecture Notes in Computer Science, vol. 7483, pp. 23–38. Springer (2012). https://doi.org/10.1007/978-3-642-33383-5_2
25. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 characteristics: Searching through a minefield of contradictions. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 7073, pp. 288–307. Springer (2011), https://doi.org/10.1007/978-3-642-25385-0_16
26. Mendel, F., Nad, T., Schläffer, M.: Collision attacks on the reduced dual-stream hash function RIPEMD-128. In: FSE. Lecture Notes in Computer Science, vol. 7549, pp. 226–243. Springer (2012), https://doi.org/10.1007/978-3-642-34047-5_14
27. Mendel, F., Nad, T., Schläffer, M.: Improving local collisions: New attacks on reduced SHA-256. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 7881, pp. 262–278. Springer (2013), https://doi.org/10.1007/978-3-642-38348-9_16

28. Mendel, F., Peyrin, T., Schläffer, M., Wang, L., Wu, S.: Improved cryptanalysis of reduced RIPEMD-160. In: ASIACRYPT(2). Lecture Notes in Computer Science, vol. 8270, pp. 484–503. Springer (2013). https://doi.org/10.1007/978-3-642-42045-0_25
29. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: Analysis of step-reduced SHA-256. In: FSE. Lecture Notes in Computer Science, vol. 4047, pp. 126–143. Springer (2006). https://doi.org/10.1007/11799313_9
30. Merkle, R.C.: One way hash functions and DES. In: Brassard, G. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 435, pp. 428–446. Springer (1989), https://doi.org/10.1007/0-387-34805-0_40
31. Mironov, I., Zhang, L.: Applications of SAT solvers to cryptanalysis of hash functions. In: SAT. Lecture Notes in Computer Science, vol. 4121, pp. 102–115. Springer (2006), https://doi.org/10.1007/11814948_13
32. Nikolic, I., Biryukov, A.: Collisions for step-reduced SHA-256. In: FSE. Lecture Notes in Computer Science, vol. 5086, pp. 1–15. Springer (2008). https://doi.org/10.1007/978-3-540-71039-4_1
33. Sanadhya, S.K., Sarkar, P.: New collision attacks against up to 24-step SHA-2. In: INDOCRYPT. Lecture Notes in Computer Science, vol. 5365, pp. 91–103. Springer (2008). https://doi.org/10.1007/978-3-540-89754-5_8
34. Stevens, M.: New collision attacks on SHA-1 based on optimal joint local-collision analysis. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 7881, pp. 245–261. Springer (2013), https://doi.org/10.1007/978-3-642-38348-9_15
35. Stevens, M., Bursztein, E., Karpman, P., Albertini, A., Markov, Y.: The first collision for full SHA-1. In: CRYPTO(1). Lecture Notes in Computer Science, vol. 10401, pp. 570–596. Springer (2017), https://doi.org/10.1007/978-3-319-63688-7_19
36. Stevens, M., Lenstra, A.K., de Weger, B.: Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 4515, pp. 1–22. Springer (2007), https://doi.org/10.1007/978-3-540-72540-4_1
37. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the hash functions MD4 and RIPEMD. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 3494, pp. 1–18. Springer (2005), https://doi.org/10.1007/11426639_1
38. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: CRYPTO. Lecture Notes in Computer Science, vol. 3621, pp. 17–36. Springer (2005), https://doi.org/10.1007/11535218_2
39. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 3494, pp. 19–35. Springer (2005), https://doi.org/10.1007/11426639_2
40. Wang, X., Yu, H., Yin, Y.L.: Efficient collision search attacks on SHA-0. In: CRYPTO. Lecture Notes in Computer Science, vol. 3621, pp. 1–16. Springer (2005), https://doi.org/10.1007/11535218_1
41. Yu, H., Bai, D.: Boomerang attack on step-reduced SHA-512. In: Inscrypt. Lecture Notes in Computer Science, vol. 8957, pp. 329–342. Springer (2014). https://doi.org/10.1007/978-3-319-16745-9_18
42. Yu, H., Wang, X.: Non-randomness of 39-step SHA-256. In: Presented at rump session of EUROCRYPT (2008)