Analysis of SHA-512/224 and SHA-512/256

Christoph Dobraunig, Maria Eichlseder, and Florian Mendel

Graz University of Technology, Austria
maria.eichlseder@iaik.tugraz.at

Abstract. In 2012, NIST standardized SHA-512/224 and SHA-512/256, two truncated variants of SHA-512, in FIPS 180-4. These two hash functions are faster than SHA-224 and SHA-256 on 64-bit platforms, while maintaining the same hash size and claimed security level. So far, no third-party analysis of SHA-512/224 or SHA-512/256 has been published. In this work, we examine the collision resistance of step-reduced versions of SHA-512/224 and SHA-512/256 by using differential cryptanalysis in combination with sophisticated search tools. We are able to generate practical examples of free-start collisions for 44-step SHA-512/224 and 43-step SHA-512/256. Thus, the truncation performed by these variants on their larger state allows us to attack several more rounds compared to the untruncated family members. In addition, we improve upon the best published collisions for 24-step SHA-512 and present practical collisions for 27 steps of SHA-512/224, SHA-512/256, and SHA-512.

Keywords: hash functions · cryptanalysis · collisions · free-start collisions · SHA-512/224 · SHA-512/256 · SHA-512 · SHA-2

1 Introduction

The SHA-2 family of hash functions is standardized by NIST as part of the Secure Hash Standard in FIPS 180-4 [17]. This standard is not superseded by the upcoming SHA-3 standard. Rather, the SHA-3 hash functions supplement the SHA-2 family. Thus, it is likely that the SHA-2 family will remain as ubiquitously deployed in the foreseeable future as it is now. Therefore, the continuous application of state-of-the-art cryptanalytic techniques for quantifying the security margin of hash functions of the SHA-2 family is of significant practical importance.

In this work, we focus on the two most recent members of the SHA-2 family, SHA-512/224 and SHA-512/256. As already observed by Gueron et al. [7], using truncated SHA-512 variants like SHA-512/256 gives a significant performance advantage over SHA-256 on 64-bit platforms due to the doubled input block size. At the same time, the shorter 256-bit hash values are more economic, compatible with existing applications, and offer the same security level as SHA-256. In addition, the resulting chop-MD [3] structure of SHA-512/224 and SHA-512/256 with is wide-pipe structure provides cryptographic benefits over the standard

Merkle-Damgård [4,16] structure by prohibiting generic attacks like Joux' multicollision attack [9], Kelsey and Kohno's herding and Nostradamus attacks [10], and Kelsey and Schneier's second preimages for long messages [11].

However, no cryptanalysis dedicated to SHA-512/224 and SHA-512/256 has been published so far. Therefore, we examine the effects of truncating the hash value of SHA-512. We show that due to this truncation, practical free-start collision for 43-step SHA-512/256 and 44-step SHA-512/224 are possible. Moreover, we improve upon the previous best collisions for 24-step SHA-512 [8,19] and show collisions for 27 steps of SHA-512, SHA-512/224, and SHA-512/256. Since all of our results are practical, we provide examples of colliding message pairs for every attack. Our results are summarized in Table 1 together with previously published collision attacks.

Table 1: Best published collision attacks on the SHA-512 family.

Hash size	Type	Steps	Complexity	Reference
all	collision	24/80	practical	[8,19]
	collision	**27**/80	**practical**	**Sect. 4.3**
	semi-free-start collision	38/80	practical	[6]
	semi-free-start collision	**39**/80	**practical**	**Sect. 4.1**
512	free-start collision	57/80	$2^{255.5}$	[13]
384	free-start collision	40/80	2^{183}	[13]
256	**free-start collision***	**43**/80	**practical**	**Sect. 4.2**
224	**free-start collision***	**44**/80	**practical**	**Sect. 4.2**

* without padding.

Related work. No dedicated cryptanalysis of SHA-512/224 or SHA-512/256 has been published so far. However, there is a number of results targeting SHA-512. The security of SHA-512 against preimage attacks was first studied by Aoki et al. [1]. They presented MITM preimage attacks on 46 steps of the hash function. This was later extended to 50 steps by Khovratovich et al. [12]. However, due to the wide-pipe structure of SHA-512/224 and SHA-512/256, these attacks do not carry over to SHA-512/224 and SHA-512/256.

The currently best known practical collision attack on the SHA-512 hash function is for 24 steps. It was published independently by Indesteege et al. [8] and by Sanadhya and Sarkar [19]. Both attacks are trivial extensions of the attack strategy of Nikolić and Biryukov [18] which applies to both SHA-256 and SHA-512. Recently, Eichlseder et al. [6] demonstrated how to extend these attacks to get semi-free-start collisions for SHA-512 reduced to 38 steps with practical complexity. Furthermore, second-order differential collisions for SHA-512 up to 48 steps with practical complexity have been shown by Yu et al. [22]. We want to note that all these practical collision attacks on SHA-512 are also applicable to its truncated variants.

Additionally, Li et al. showed in [13] that particular preimage attacks on SHA-512 can also be used to construct free-start collision attacks for the step-reduced hash function and its truncated variants. They show a free-start collision for 57-step SHA-512 and 40-step SHA-384. Both attacks are only slightly faster than the respective generic attacks.

Outline. The remainder of the paper is organized as follows. We describe the design of the SHA-2 family in Sect. 2. Then, we briefly explain our attack strategy and discuss the choice of suitable starting points for our attacks in Sect. 3. The actual attacks on step-reduced SHA-512/224 and SHA-512/256 are presented in Sect. 4.

2 Description of SHA-512 and other SHA-2 variants

The SHA-2 family of hash functions is specified by NIST as part of the Secure Hash Standard (SHS) [17]. The standard defines two main algorithms, SHA-256 and SHA-512, with truncated variants SHA-224 (based on SHA-256) and SHA-512/224, SHA-512/256, and SHA-384 (based on SHA-512). In addition, NIST defines a general truncation procedure for arbitrary output lengths up to 512 bits. Below, we first describe SHA-512, followed by its truncated variants SHA-512/224 and SHA-512/256 that this paper is focused on. Finally, the main differences to SHA-256 and SHA-224 are briefly discussed.

SHA-512. SHA-512 is an iterated hash function that pads and processes the input message using t 1024-bit message blocks m_j. The 512-bit hash value is computed using the compression function f:

$$\begin{aligned} h_0 &= \text{IV}, \\ h_{j+1} &= f(h_j, m_j) \qquad \text{for } 0 \le j < t. \end{aligned}$$

The hash output is the final 512-bit chaining value h_t.

In the following, we briefly describe the compression function f of SHA-512. It basically consists of two parts: the message expansion and the state update transformation. A more detailed description of SHA-512 is given by NIST [17].

We use $+$ (or $-$) to denote addition (or subtraction) modulo 2^{64}; $\oplus$ (or $\wedge$) is bitwise exclusive-or (or bitwise and) of 64-bit words, and $\ggg n$ (or $\gg n$) denotes rotate-right (or shift-right) by n bits.

Padding and message expansion. The message expansion of SHA-512 splits each 1024-bit message block into 16 64-bit words M_i, $i = 0, \ldots, 15$, and expands these into 80 expanded message words W_i as follows:

$$W_i = \begin{cases} M_i & 0 \leq i < 16, \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} & 16 \leq i < 80. \end{cases} \quad (1)$$

The functions $\sigma_0(x)$ and $\sigma_1(x)$ are given by

$$\sigma_0(x) = (x \ggg 1) \oplus (x \ggg 8) \oplus (x \gg 7),$$
$$\sigma_1(x) = (x \ggg 19) \oplus (x \ggg 61) \oplus (x \gg 6).$$

State update transformation. We use the alternative description of the SHA-512 state update by Mendel et al. [14], which is illustrated in Fig. 1.

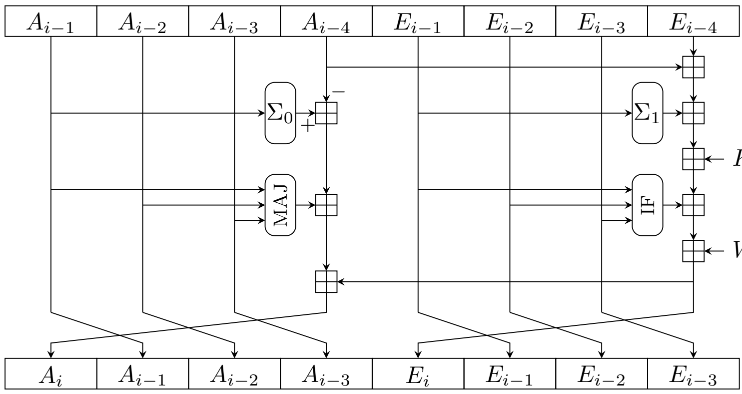


Fig. 1: The state update transformation of SHA-512.

The state update transformation starts from the previous 512-bit chaining value $h_j = (A_{-1}, \ldots, A_{-4}, E_{-1}, \ldots, E_{-4})$ and updates it by applying the step functions 80 times. In each step $i = 0, \ldots, 79$, one 64-bit expanded message word W_i is used to compute the two state variables E_i and A_i as follows:

$$E_i = A_{i-4} + E_{i-4} + \Sigma_1(E_{i-1}) + \mathrm{IF}(E_{i-1}, E_{i-2}, E_{i-3}) + K_i + W_i, \quad (2)$$
$$A_i = E_i - A_{i-4} + \Sigma_0(A_{i-1}) + \mathrm{MAJ}(A_{i-1}, A_{i-2}, A_{i-3}). \quad (3)$$

For the definition of the step constants K_i, we refer to the standard document [17]. The bitwise Boolean functions IF and MAJ used in each step are defined by

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

and the linear functions Σ_0 and Σ_1 are defined as follows:

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

After the last step of the state update transformation, the previous chaining value is added to the output of the state update (Davies-Meyer construction). The result of this feed-forward sum is the chaining value h_{j+1} for the next message block m_{j+1} (or the final hash value h_t):

$$h_{j+1} = (A_{79} + A_{-1}, \ldots, A_{76} + A_{-4}, E_{79} + E_{-1}, \ldots, E_{76} + E_{-4}). \qquad (4)$$

SHA-512/256 and SHA-512/224. These truncated variants of SHA-512 differ only in their initial values and a final truncation to 256 or 224 bits, respectively. The rest of the algorithmic description remains exactly the same. The message digest of SHA-512/256 is obtained by omitting the output words $E_{79}+E_{-1}$, $E_{78}+E_{-2}$, $E_{77}+E_{-3}$, and $E_{76}+E_{-4}$ of the last compression function call. SHA-512/224 additionally omits the 32 least significant bits of $A_{76} + A_{-4}$.

SHA-256 and SHA-224. SHA-256 and SHA-512 are closely related. Thus, we only point out properties of SHA-256 which differ from SHA-512:

- The wordsize is 32 instead of 64 bits.
- IV and K_i are the 32 most significant bits of the respective SHA-512 value.
- The step function is applied 64 instead of 80 times.
- The linear functions $\sigma_0, \sigma_1, \Sigma_0$ and Σ_1 use different rotation values.

SHA-224 is a truncated variant of SHA-256 with different IV, in which the output word $E_{60} + E_{-4}$ is omitted.

3 Attack strategy

Starting from the ground-breaking results of Wang et al. [20,21], the search techniques used for practical collisions have been significantly improved, hitting their current peak in the attacks on SHA-256 [2,15] and SHA-512 [6,22]. In spite of all achieved improvements, the top-level attack strategy has remained essentially the same. At first, a suitable starting point for the search must be determined to define the search space and hopefully make the ensuing search process feasible. The search itself usually involves two phases: The search for a suitable differential characteristic, and the message modification phase to determine a collision-producing message pair for this characteristic. The search for this characteristic and message pair can either be done by hand or, for more complex functions like SHA-2, using an automatic search tool. We use a heuristic search tool based on a guess-and-determine strategy, which we briefly describe in Sect. 3.1. Afterwards, we discuss the choice of suitable starting points in Sect. 3.2.

3.1 Guess-and-determine search tool

To search for differential characteristics and colliding message pairs, we use an automatic search tool, which implements a configurable heuristic guess-and-determine search strategy. Roughly, the tool is partitioned into two separate, but closely interacting parts: The representation of the analyzed cryptographic primitive and the search procedure.

Representation. The tool internally represents differences at bit level, allowing to store all possible stages from a completely unrestricted bit over signed differences down to exact values. Thus, the same tool can be used in the search for a characteristic and in the search for a message pair. The conditions are grouped in words representing the internal variables of the hash function. These words can then be connected with any operations (typically bitwise functions or modular additions) to define the hash function.

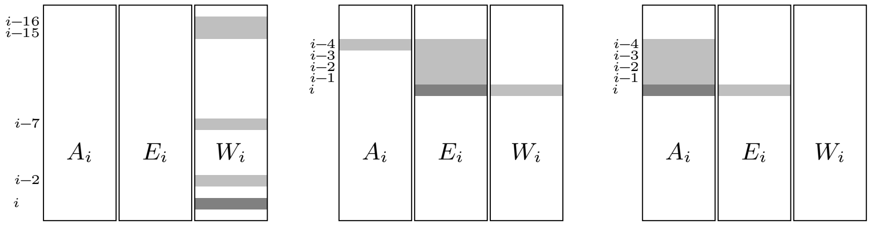
Search. The search procedure uses the bitwise conditions as variables, and attempts to find a solving assignment with the help of a heuristic guess-and-determine strategy [5], similar to SAT solvers. The following steps are repeated until a solution is found:

- **Guess**: Pick a bit and guess its value (e.g., no difference, or a specific assignment).
- **Determine**: The previous guess influences other connected bit conditions. Determine these effects, which might result in further refinement of other bit conditions, or a contradiction.
- **Backtrack**: If a contradiction is detected, resolve this conflict by undoing previous guesses and replacing them with other choices.

This simple approach alone is not sufficient to go through the whole search space, so numerous refinements have been proposed to fine-tune this method. These include the detection of two-bit conditions [14], backtracking strategies, and a look-ahead approach to guide the search [6]. Additionally, SHA-2-specific heuristics and strategies [14,15] have been proposed, deciding which parts of the state to guess with higher priority.

3.2 Finding starting points for SHA-2

To model SHA-2 as a satisfiability problem for the search tool, we need to introduce suitable intermediate variables. Based on the alternative description from Sect. 2, we only use the words A_i and E_i of the state, plus the words W_i of the message expansion. Fig. 2 illustrates the update rules for A, E and W by highlighting the input words for updating each word: Each row represents one of the 80 step iterations, with its three state words A_i, E_i, and W_i.



(a) Updating W_i via (1) (b) Updating E_i via (2) (c) Updating A_i via (3)

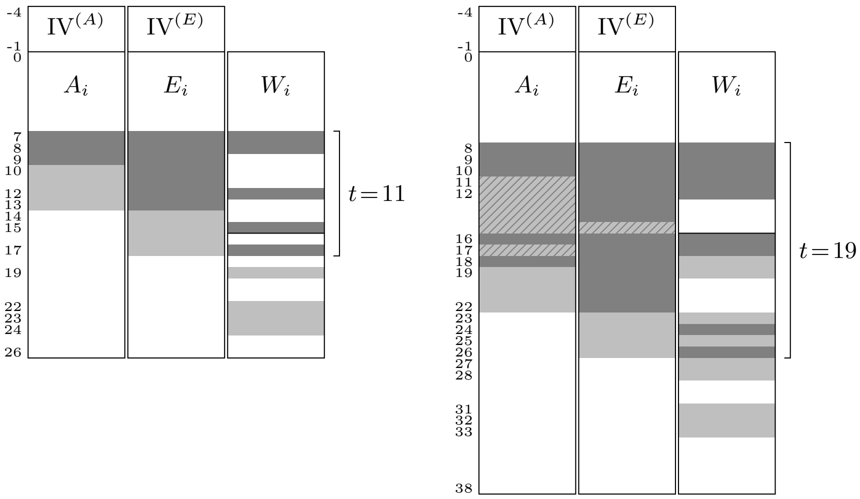
Fig. 2: Update rules to compute A_i, E_i, and W_i (■) from other state words (■).

Local collisions. All our results are based on "local collisions" in the message expansion: by carefully selecting (expanded) message words in the middle steps so that the differences can cancel out in as many consecutive steps as possible in the forward and backward expansion, i.e., the first and last few expanded message words contain no differences. The t middle steps with differences can induce differences in the A_i and E_i words. However, the W_i words can be used to achieve zero difference in the last 4 of the t words E_i, and in the last 8 of the t words A_i. This is necessary to obtain words with zero difference in the very last 4 steps of the state update and thus in the output chaining value.

As an example, the starting point for the 27-step collisions for SHA-256 [14] allows differences in expanded message words W_7, W_8, W_{12}, W_{15}, and W_{17}, as well as state words $E_7, \ldots, E_{13}$ and $A_7, \ldots, A_{10}$. The exact bitwise signed differences are chosen during the search such that any potential differences in $W_{19}, W_{22}, W_{23}, W_{24}$, as well as $E_{14}, \ldots, E_{17}$ and $A_{10}, \ldots, A_{13}$ cancel out. The resulting starting point is illustrated in Fig. 3a. We show in Sect. 4.3 how the same starting point can be used for SHA-512.

The semi-free-start collision starting point covering the most steps so far is for 38 steps of SHA-256 [15] and SHA-512 [6], with a local collision spanning $t = 18$ steps. Considering the large number of steps, the number of expanded message words with differences and cancellations is remarkably low: only 6 words with differences, and 6 words imposing cancellation conditions.

To find candidates for a higher number of steps, we enumerated all possible selections of active message words (more precisely, of some $t \leq 20$ intermediate expanded message words, the "core words" of the local collision) and investigated the forward and backward expansion under certain assumptions: the t core words are chosen freely, according to the message expansion rule; in the forward and backward expansion, if at least 2 of the input words have differences, they are assumed to cancel out, while a single input word with difference never cancels out. Criteria for selecting suitable candidates then include a low number t of spanned steps and a low number of required cancellation constraints. The best (consistent) result for 39 steps, spanning $t = 19$ steps with 9 cancellations, is given in Fig. 3b.



(a) 27-step collision of SHA-256 [14] and SHA-512 (Sect. 4.3).

(b) 39-step semi-free-start collision of SHA-512 (Sect. 4.1).

Fig. 3: SHA-2 starting points: Words with differences ■ and cancellations ■, ▨.

Semi-free-start collisions and collisions. The discussed starting points are targeted to find semi-free-start collisions, that is, different messages m, m' and an IV h_0 such that $f(h_0, m) = f(h_0, m')$. However, they can also be used for hash function collisions with the original IV h_0 by trading the freedom of the IV for freedom in the message words.

In order to find hash function collisions, the first few message words W_i must retain sufficient freedom (i.e., they should not be constrained by conditions from the message expansion for cancelling differences) to allow to match the correct IV value. Ideally, this means that the first 8 message words $W_0, \ldots, W_7$ are free of any conditions (no differences, but also not constrained by conditions from other message words connected via the message expansion). If the W_i differences are sparse enough overall, it can also be sufficient to have at least 5 words $W_0, \ldots, W_4$ free of conditions by providing the remaining freedom with a two-block approach [15].

The starting points of Fig. 3a and Fig. 3b both have at least 7 message words free of differences in the beginning. However, the local collision shown in Fig. 3b spans over $t = 19$ steps. Thus, the first message words are constrained by many conditions, leaving not enough freedom to match the correct IV. In contrast, the 11-step local collision shown in Fig. 3a provides enough freedom in the first 7 message words to be used in a single-block collision attack [14].

4 Collision attacks for truncated SHA-512 variants

The hash functions SHA-512/224 and SHA-512/256 differ from SHA-512 in their IV and a final processing step, which truncates the 512-bit state to 224 or 256 bits, respectively. Consequently, the semi-free-start collisions demonstrated for SHA-512 [6] are also valid for these truncated versions (since the IV is non-standard anyway in this attack scenario). In this section, we first improve these results by providing 39-step semi-free-start collisions for SHA-512 and its variants. We then extend this result to free-start collisions for 43-step SHA-512/256 and 44-step SHA-512/224. By free-start collisions, we mean two messages m, m' and two IVs h_0, h'_0 such that the hash values of m (under IV h_0) and m' (under IV h'_0) collide. Note that free-start collisions are not equivalent to collisions of the compression function for truncated SHA-2 versions, since the truncated output bits of the last compression function call may contain differences. Additionally, we present collisions for 27 steps of SHA-512, SHA-512/224, and SHA-512/256.

4.1 Semi-free-start collisions

We use the 39-step starting point from Fig. 3b. Previous work showed that sparse differences particularly in the A_i words are essential for the success probability of the message modification phase. For this reason, we additionally require that in 6 words between A_8 and A_{18}, namely $A_{11}, A_{12}, A_{13}, A_{14}, A_{15}$, and A_{17}, differences also cancel out. The five consecutive zero-difference words in A_i also force E_{15} to zero difference. These additional requirements are already marked in Fig. 3b (hatched area).

The first task for the search procedure with the solving tool is to fix a suitable signed characteristic. Compared to the previously published 38-step SHA-512 semi-free-start collision [6], the local collision for our starting point spans 19 steps (compared to previously 18) and has 9 (previously 6) active expanded message words. Cancellations are also required in 9 (previously 6) expanded message words. This increases the necessity for very sparse differences in A_i and W_i in steps 16–26. For this reason, we require a single-bit difference in W_{26}, W_{17} and A_{18}, and very low Hamming weights for the other words. We finally found a characteristic with at most two active bits in almost all words of A_i and W_i (except $A_9, A_{10}, W_{11}, W_{12}$).

After the characteristic is fixed, we need to find a complying message pair. We start by guessing the dense parts in A_i and E_i, hoping that the sparser conditions in the later steps are fulfilled probabilistically. Since the dense parts are already almost fully determined by the characteristics and the sparse parts pose only so few conditions, a message pair is easily found. The result is a semi-free-start collision valid for all SHA-512 variants. We give an example in Appendix A in Table 4a.

4.2 Free-start collisions

Free-start collisions are a generalization of semi-free-start collisions, so the 39-step results obtained in the previous section give a first result for SHA-512/224 and

SHA-512/256. However, we can take advantage of the truncated output bits to add several more steps. If we add another step in the beginning or in the end, the existing difference pattern remains unchanged, but there will be differences in the word W_0 (computable via backward expansion, which includes $W_{i+9} = W_9$, the previous W_8 from Fig. 3b) or in the new word W_{39} (via the normal forward expansion, which includes $W_{39-15} = W_{24}$), respectively. These, in turn, can imply differences in E_{-4} or in A_{39} and E_{39}, which translates to differences in the IV (turning semi-free-start into free-start results, and included in the hash value via the feed-forward) or directly in the compression function output, respectively.

The advantage of adding steps in the beginning is that it is possible to limit the additional differences in the state update words to E, and keep A free of new differences. Any differences in $E_{-1}, \ldots, E_{-4}$ will be added to the compression function output with the final feed-forward, but the corresponding words of the result are truncated, so the hash outputs still collide.

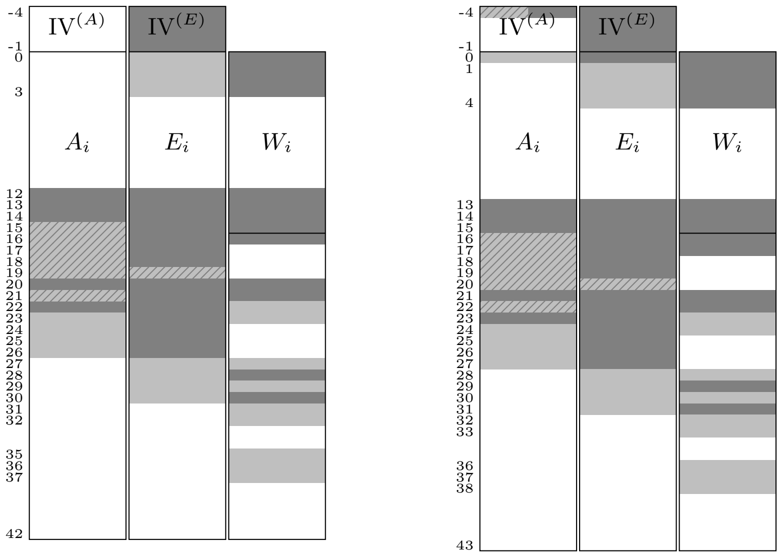
Free-start collisions for 43-step SHA-512/256. Since SHA-512/256 truncates the last 4 output words of the compression function call ($E_{79} + E_{-1}$, $E_{78}+E_{-2}$, $E_{77}+E_{-3}$, and $E_{76}+E_{-4}$), differences in $E_{-1}, \ldots, E_{-4}$ are acceptable for a free-start collision. This observation allows us to add 4 additional steps in the beginning of the 39-step starting point from Fig. 3b. Shifting the characteristic "downwards" by 4 steps causes the previous message words $W_{12}, \ldots, W_{15}$ to turn into new expanded message words $W_{16}, \ldots, W_{19}$; in particular, this affects the difference in the previous word W_{12}. To determine a compatible difference pattern for the new first 4 words, the message expansion can be computed backwards from the new words $W_4, \ldots, W_{19}$ via

$$W_i = W_{i+16} - \sigma_1(W_{i+14}) - W_{i+9} - \sigma_0(W_{i+1}).$$

It turns out that all 4 new words will contain differences (W_3 from $W_{3+9} = W_{12}$; W_2 from $W_{2+1} = W_3$ and $W_{2+14} = W_{16}$; W_1 from $W_{1+1} = W_2$ and $W_{1+14} = W_{15}$; and W_0 from $W_{0+1} = W_1$, $W_{0+14} = W_{14}$ and $W_{0+16} = W_{16}$). However, similar to steps 27–30, the state words A_i and E_i can be kept free of differences for 4 steps. To achieve this, the search tool needs to find differences in the IV words $E_{-4}, \ldots, E_{-1}$ to cancel out those in $W_0, \ldots, W_3$ when computing $E_0, \ldots, E_3$. The resulting starting point is given in Fig. 4a.

For the search procedure with the solving tool, we fixed the signed differences of steps 12–30 to the same values as the 39-step SHA-512 semi-free-start collision of Sect. 4.1. Then, to complete the characteristic, we first search for a valid solution for the dense part of the middle steps (A_i and E_i in steps 13–16, and E_i in steps 17–27), and finally fix the corresponding message words W_i in steps 13–17, which determines the complete state, including the dense differences in the prepended steps and IV.

The search only takes seconds on a standard computer; an example for a free-start collision is given in Appendix A in Table 3a.



(a) 43-step SHA-512/256. (b) 44-step SHA-512/224.

Fig. 4: Potential free-start starting points (differences ■ and cancellations ■, ▨).

Free-start collisions for 44-step SHA-512/224. A very similar strategy can be employed to extend the previous 43-step free-start collision by another step for SHA-512/224. Prepending an additional step shifts the difference of previous word E_{-1} to E_0, which in turn requires a cancellation in A_0 and a difference in A_{-4}, as illustrated in Fig. 4b. However, only the least significant 32 bits of the corresponding compression function output word are truncated. Furthermore, this output word is computed from A_{-4} via modular addition, so even differences only in the lower 32 bits can possibly cause differences in the untruncated output bits.

Fortunately, the underlying characteristic of signed differences as used for the 39-step SHA-512 semi-free-start collision is well compatible with our constraints: The difference in A_{-4} needs to cancel that in W_4 in a modular addition (via E_0, by equations (3) and (2) or Fig. 2, since all other involved words have zero difference). This difference of W_4, in turn, is dictated by that in W_{13} (by the update rule for W_{20}, where again all other involved words have zero difference). None of these equalities involves any of the bitwise functions σ, Σ, MAJ or IF. Thus, the modular difference in A_{-4} must be the same as that in W_{13}, which is already fixed by the underlying characteristic to a modular difference of $+32$. Written as bitwise differences, this will translate to a single-bit difference (in the sixth least significant bit) with probability $\frac{1}{2}$ (which does not carry over to the untruncated bits of the final output with overwhelming probability). Indeed, the example for a free-start collision given in Appendix A in Table 2a only displays this single-bit difference in A_{-4} (and no carries in the output bits).

4.3 Collisions

So far, the best practical collisions found for SHA-512 are those for 24 steps, proposed independently by Sanadhya and Sarkar [19] and Indesteege et al. [8], together with 24-step collisions for SHA-256. While the results for SHA-256 have since been improved to 27 [14], 28 [15] (both practical), and finally 31 steps [15] (theoretical attack with almost practical complexity), no such improvements have been proposed for SHA-512 so far. The main reason for this seems to be the doubling in state size from SHA-256 to SHA-512; this larger search space increases the difficulty of the problem for the search tools.

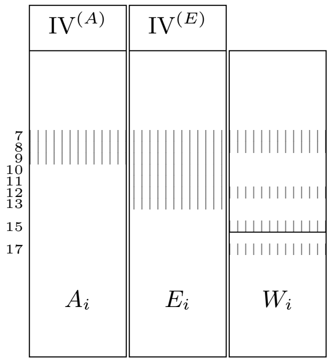
Starting point for SHA-512. Since the message expansion is essentially the same for all SHA-2 variants (except for different word sizes and rotation values, of course), the SHA-256 starting points can theoretically also be used for SHA-512. However, the resulting search complexity is different. For our results, we used the 27-step starting point (based on a local collision over the $t = 11$ steps 7–17), as illustrated in Fig. 3a. Just as the 39-step semi-free-start starting point (Fig. 3b), it requires that differences cancel in E in 4 of the t steps ($E_{14}, \ldots, E_{17}$) and in A in the 4 previous steps ($A_{10}, \ldots, A_{13}$), as well as in several steps of the message expansion.

Finding a solution from this starting point requires significantly more effort than for SHA-256. Of course, we also tried to expand our search to the closely related 28-step starting point, which adds an additional step in the beginning of the 27-step version. However, with the additional constraints imposed on the message expansion by this added step we could not find any suitable (reasonably sparse) characteristics.

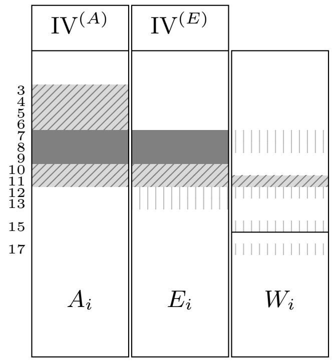
In contrast to the results from Sect. 4.2, since the IV needs to exactly match the original IV, we were not able to take advantage of the final truncation to simplify the search process, or add additional steps. We first search a characteristic for SHA-512, and then try to use it to match the different IVs for SHA-512/224 and SHA-512/256.

Search strategy. The search progresses in several stages, as illustrated in Fig. 5:

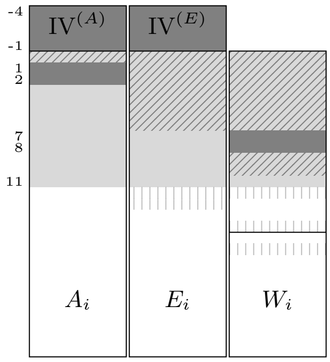
1. **Fix signed characteristic**:
   (a) **Find candidate characteristic** (Fig. 5a): First fix the signed differences of the message expansion W (5 words) and state update A (3 words). Since the word W_{17} poses conditions on the first few message words, whose freedom we will later need to match the IV, we focus on keeping its signed difference as sparse as possible, with only few difference bits. With much lower priority, also determine the differences in the state update words E (7 words) to complete the signed characteristic. The characteristic is very dense in E, but this only has limited influence on the success of the IV matching phase.



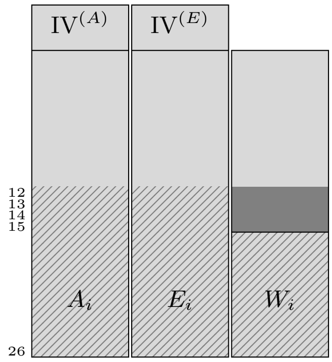
(a) Stage 1a (characteristic).



(b) Stage 1b (dense part).



(c) Stage 2a (match IV).



(d) Stage 2b (sparse part).

Fig. 5: Stages of the 27-step collision search (guessed values ■ and differences ◫, derived values ▨, and previously fixed values ■ and differences ◫).

(b) **Verify dense parts** (Fig. 5b): Fully determine the values of A and E in the densest steps 7–9 to verify the validity of the candidate characteristic. If necessary, fix any remaining free bits of A and E in steps 10–11. This fully determines $A_3, \ldots, A_{11}$, $E_7, \ldots, E_{11}$ and W_{11}.

To maneuver the search process in the large search space and detect contradictions as soon as possible, we need to apply the look-ahead strategies previously employed for semi-free-start collisions on SHA-512 [6] in this stage (with 16 look-ahead candidates per guess).

2. **Message modification to match IV**: Starting from the best signed characteristics of the previous stage, with the correct IV inserted, find a solution message pair step by step:

(a) **Match IV** (Fig. 5c): Fix the values in the more difficult, heavily constrained words first (W_{10}, W_9, W_8, W_7). Choosing W_{10} and W_9 also determines A_2 and A_1 (via E_6 and E_5). Together with W_7, W_8, and the IV, this determines all values in steps 0–11.

(b) **Finalize message for sparse parts** (Fig. 5d): choosing the 4 remaining message words $W_{12}, \ldots, W_{15}$ allows to satisfy the remaining, sparse parts of the characteristic in steps 12–26 with high probability.

Unlike the other stages, guesses are not made randomly here, but systematically word-by-word. Since most conditions are from modular additions, we always start from the least significant bits and proceed towards the more significant bits. This last stage needs to be repeated for each IV separately, which takes some hours on a single CPU per target IV.

Results. Our results for collisions for 27-step SHA-512/224, SHA-512/256, and SHA-512 are given in Appendix A in Tables 2b, 3b, and 4b, respectively.

Acknowledgments. This research (or a part of this research) is supported by Cryptography Research and Evaluation Committee (CRYPTREC) and by the Austrian Research Promotion Agency (FFG) and the Styrian Business Promotion Agency (SFG) under grant number 836628 (SeCoS).

A Examples

An example for the semi-free-start collisions of Sect. 4.1 is given in Table 4a. Results for the free-start collisions of Sect. 4.2 are given in Tables 2a and 3a, and for the collisions of Sect. 4.3 in Tables 2b, 3b, and 4b.

Table 2: Results for SHA-512/224.

(a) Example of a free-start collision for 44 steps of SHA-512/224.

h_0	fef65b64d3694995	959fbfb82ed84eb1	1d9e855642e62ef2	335cc6d027695d91
	921d197e5cfa2803	e26c6eb26163a692	9ff3cf4d26f1de78	5323942861d9139a
h_0^*	fef65b64d3694995	959fbfb82ed84eb1	1d9e855642e62ef2	335cc6d027695db1
	a712860cdcfa1ff8	470749bbf7628f44	20cdfd694df67216	8e07b5fa2c7fedf0
Δh_0	0000000000000000	0000000000000000	0000000000000000	0000000000000020
	350f9f72800037fb	a56b2709960129d6	bf3e32246b07ac6e	dd2421d24da6fe6a
m	7a19df6089d00684	03ed2a0d0c29e00e	36c91e35f681fbb8	bb2b47428aeff294
	dce94ccc981d39a3	44230f73cf56d9ef	e9d46b26b44950c8	550bed4b9419741c
	58a98894206e00de	f3448a6f761d384d	9ae59f3a3bcc5bba	ece85d5c77be431b
	6e3cf817e9376cc7	b74a2a43c0b96c93	7c5b51d6fe2a0c26	5a9868e5bf2e422d
m^*	5e031bbe28b2d027	ded424ef85255cc3	ad2f514be0830c1f	a635dab40aeffa9f
	dce94ccc981d3983	44230f73cf56d9ef	e9d46b26b44950c8	550bed4b9419741c
	58a98894206e00de	f3448a6f761d384d	9ae59f3a3bcc5bba	ece85d5c77be431b
	6e3cf817e9376cc7	b74a2a43c0b96cb3	5c5b51d6fe2a0c36	5a9868e5bf2e420d
Δm	241ac4dea162d6a3	dd390ee2890cbccd	9be64f7e1602f7a7	1d1e9df68000080b
	0000000000000020	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000020	2000000000000010	0000000000000020
h_1	e309edf68f4d89b8	5c356e0359eb0dab	76b4a45ec3c2cd25	8bd0955d

(b) Example of a collision for 27 steps of SHA-512/224.

m	20dbf13a352116a9	295506e205afd435	abfe4826742c1a1a	279f07c7813dd9be
	47da77c701a98858	25aec1349d486501	37a992a15616ea31	e2b122ecf19e90d3
	2fff6025dc03dd67	032c261d740f459e	2e2599bd6e7e74df	d490bd22815eb494
	72fedf1f607df6e3	87fc91fcfb7397fd	e647b1b499eee17f	2dff8e493cbc8a4c
m^*	20dbf13a352116a9	295506e205afd435	abfe4826742c1a1a	279f07c7813dd9be
	47da77c701a98858	25aec1349d486501	37a992a15616ea31	5cc1250cb19e90d3
	203fdfe5dc03dd66	032c261d740f459e	2e2599bd6e7e74df	d490bd22815eb494
	f0bc01167075f6eb	87fc91fcfb7397fd	e647b1b499eee17f	d3f8fe713d7c8a4c
Δm	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	be7007e040000000
	0fc0bfc000000001	0000000000000000	0000000000000000	0000000000000000
	8242de0910080008	0000000000000000	0000000000000000	fe07703801c00000
h_1	65b11e66e48da563	1b70d12da92e2dba	8f338768bb95601b	60b995bb

Table 3: Results for SHA-512/256.

(a) Example of a free-start collision for 43 steps of SHA-512/256.

h_0	159b52516f10f30d	546b2042f240afee	f25339b24c441edf	d62c698666558242
	e5a9e39861fbd81d	d2138eacc20d5224	a332c16df23609fb	73f78341dfd7a4e5
h_0^*	159b52516f10f30d	546b2042f240afee	f25339b24c441edf	d62c698666558242
	e5a9e39861fbd83d	72e259ce420d5a0f	4db37906cc361264	ae579d9e0275b446
Δh_0	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000020	a0f1d7628000082b	ee81b86b3e001b9f	dda01edfdda210a3
m	cfbec86f1cf6821e	dd3343c25aad835a	2a08612b753f3d6b	b328d40d2c624ef7
	b3e51f8a3a63bd6f	4abdf96375bbf609	a8c5c1f784672e86	a78e2aa625830d4b
	169dcb5039bf3d9f	fbcc43ffebd8ae47	1b3eaefccf5c6a46	f668a2a728851b4e
	374601ea44422bdb	2ca290d26a23a02f	6685babbfdcb5e22	e000111457201fd4
m^*	ee37d77210586a56	b2a4122800ad72cf	89399609f53f3560	b328d40d2c624ed7
	b3e51f8a3a63bd6f	4abdf96375bbf609	a8c5c1f784672e86	a78e2aa625830d4b
	169dcb5039bf3d9f	fbcc43ffebd8ae47	1b3eaefccf5c6a46	f668a2a728851b4e
	374601ea44422bfb	0ca290d26a23a03f	6685babbfdcb5e02	e07e151457202055
Δm	21891f1d0caee848	6f9751ea5a00f195	a331f7228000080b	0000000000000020
	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000020	2000000000000010	0000000000000020	007e040000003f81
h_1	1d7041bbbffa676a	03d8c440d9246b9d	20ce2d17c5b0b2c4	7e6e4d33a7f54afd

(b) Example of a collision for 27 steps of SHA-512/256.

m	306b0c2ebe7c1341	c8b55d4df1c5f4fe	b91a173aeceb818a	33b5977f9b46e58b
	6c6d5a4f87f1364f	1b7e33249d4acf4f	b7f784ecdcaefc1f	a33edafe7afc0452
	dfc0200932c2b9df	faec7d05e3518e56	ec2e19a7ee867396	d490bd22815eb494
	72fedf1f887df303	f95891f08483da25	c327d0afa2c4f902	2c5f0c0806a4e298
m^*	306b0c2ebe7c1341	c8b55d4df1c5f4fe	b91a173aeceb818a	33b5977f9b46e58b
	6c6d5a4f87f1364f	1b7e33249d4acf4f	b7f784ecdcaefc1f	1d4edd1e3afc0452
	d0009fc932c2b9de	faec7d05e3518e56	ec2e19a7ee867396	d490bd22815eb494
	f0bc01169875f30b	f95891f08483da25	c327d0afa2c4f902	d2587c300764e298
Δm	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	be7007e040000000
	0fc0bfc000000001	0000000000000000	0000000000000000	0000000000000000
	8242de0910080008	0000000000000000	0000000000000000	fe07703801c00000
h_1	fcba5c8faf05fd68	c676b8f17b5daae3	6233801174b7fd01	0ff72ab4a869c54f

Table 4: Results for SHA-512.

(a) Example of a semi-free-start collision for 39 steps of SHA-512.

h_0	eccf3da189dd9668	b1ec21a4fd53b8d8	609ce4465f772770	adf4e7738e2978f6
	8edd237ea50eebc9	231b3af0102a926d	db45e613e8d2fd52	ad384433420073f6
m	a0ec9872cfffe63c	df5c6a2b59f4c453	f2bea3763fc8fa7a	6a47e8ff0a995116
	fa59232e8b617048	4c9690984c084498	28bee8f5701eab16	8d57686ecbdce623
	3879318f901ff782	72644b0ca55a6142	6cb281dab11480b4	4a8198441f401ff2
	5ffd956ed11a2b5f	9a640988d68287d3	74942df792f2637f	b2819dc61f772d4f
m^*	a0ec9872cfffe63c	df5c6a2b59f4c453	f2bea3763fc8fa7a	6a47e8ff0a995116
	fa59232e8b617048	4c9690984c084498	28bee8f5701eab16	8d57686ecbdce623
	3879318f901ff7a2	52644b0ca55a6152	6cb281dab1148094	4aff9c441f402073
	6001956ed11a2a5f	9a640988d68287d3	74942df792f2637f	b2819dc61f772d4f
Δm	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000020	2000000000000010	0000000000000020	007e040000003f81
	3ffc000000000100	0000000000000000	0000000000000000	0000000000000000
h_1	3aa73bfae7b82789	711f2024cf0f636e	0c6965f707279a53	8227fba8617aa955
	fdd9e2ca8c4d0038	57db244560d7b70b	08ec5698343353c0	9e9b739ee307ea92

(b) Example of a collision for 27 steps of SHA-512.

m	537e7a4986aa2fce	11206ad0306c752b	90124a9e1c9b0ce2	8c14e0356fd26f5f
	fd3ef90ea3e4366f	35d8c2ba58abd92f	b23e476632eca1fd	e2b122ef46649b73
	dfc020070e628f37	7acf74d1d1007558	6c6359a6fe7fe2f0	d490bd22815eb494
	72fedf1f807df6f3	a8585af19b6dd9d1	3d2053b0c295522b	2d970e0e52a49081
m^*	537e7a4986aa2fce	11206ad0306c752b	90124a9e1c9b0ce2	8c14e0356fd26f5f
	fd3ef90ea3e4366f	35d8c2ba58abd92f	b23e476632eca1fd	5cc1250f06649b73
	d0009fc70e628f36	7acf74d1d1007558	6c6359a6fe7fe2f0	d490bd22815eb494
	f0bc01169075f6fb	a8585af19b6dd9d1	3d2053b0c295522b	d3907e3653649081
Δm	0000000000000000	0000000000000000	0000000000000000	0000000000000000
	0000000000000000	0000000000000000	0000000000000000	be7007e040000000
	0fc0bfc000000001	0000000000000000	0000000000000000	0000000000000000
	8242de0910080008	0000000000000000	0000000000000000	fe07703801c00000
h_1	d838f1d2ae4bf185	3fc837ae9bbc28d4	6b2f2977f58a9697	99c48839f0e8bdca
	c9c0a86fed1d921a	2f823b1fa1913751	3ba170b902c6da30	9c4e5807be51a7e7

References

1. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for step-reduced SHA-2. In: Matsui, M. (ed.) Advances in Cryptology – ASIACRYPT 2009. LNCS, vol. 5912, pp. 578–597. Springer (2009)
2. Biryukov, A., Lamberger, M., Mendel, F., Nikolic, I.: Second-order differential collisions for reduced SHA-256. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. LNCS, vol. 7073, pp. 270–287. Springer (2011)
3. Coron, J., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer (2005)
4. Damgård, I.: A design principle for hash functions. In: Brassard, G. (ed.) Advances in Cryptology – CRYPTO '89. LNCS, vol. 435, pp. 416–427. Springer (1989)
5. De Cannière, C., Rechberger, C.: Finding SHA-1 characteristics: General results and applications. In: Lai, X., Chen, K. (eds.) Advances in Cryptology – ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer (2006)
6. Eichlseder, M., Mendel, F., Schläffer, M.: Branching heuristics in differential collision search with applications to SHA-512. In: Cid, C., Rechberger, C. (eds.) Fast Software Encryption – FSE 2014. LNCS, Springer (2014), in press
7. Gueron, S., Johnson, S., Walker, J.: SHA-512/256. In: Latifi, S. (ed.) Information Technology: New Generations – ITNG 2011. pp. 354–358. IEEE Computer Society (2011)
8. Indesteege, S., Mendel, F., Preneel, B., Rechberger, C.: Collisions and other non-random properties for step-reduced SHA-256. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) Selected Areas in Cryptography – SAC 2008. LNCS, vol. 5381, pp. 276–293. Springer (2009)
9. Joux, A.: Multicollisions in iterated hash functions. Application to cascaded constructions. In: Franklin, M.K. (ed.) Advances in Cryptology – CRYPTO 2004. LNCS, vol. 3152, pp. 306–316. Springer (2004)
10. Kelsey, J., Kohno, T.: Herding hash functions and the Nostradamus attack. In: Vaudenay, S. (ed.) Advances in Cryptology – EUROCRYPT 2006. LNCS, vol. 4004, pp. 183–200. Springer (2006)
11. Kelsey, J., Schneier, B.: Second preimages on n-bit hash functions for much less than 2^n work. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 474–490. Springer (2005)
12. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for preimages: Attacks on Skein-512 and the SHA-2 family. In: Canteaut, A. (ed.) Fast Software Encryption – FSE 2012. LNCS, vol. 7549, pp. 244–263. Springer (2012)
13. Li, J., Isobe, T., Shibutani, K.: Converting meet-in-the-middle preimage attack into pseudo collision attack: Application to SHA-2. In: Canteaut, A. (ed.) Fast Software Encryption – FSE 2012. LNCS, vol. 7549, pp. 264–286. Springer (2012)
14. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 characteristics: Searching through a minefield of contradictions. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. LNCS, vol. 7073, pp. 288–307. Springer (2011)
15. Mendel, F., Nad, T., Schläffer, M.: Improving local collisions: New attacks on reduced SHA-256. In: Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology – EUROCRYPT 2013. LNCS, vol. 7881, pp. 262–278. Springer (2013)
16. Merkle, R.C.: One way hash functions and DES. In: Brassard, G. (ed.) Advances in Cryptology – CRYPTO '89. LNCS, vol. 435, pp. 428–446. Springer (1989)

17. National Institute of Standards and Technology: FIPS PUB 180-4: Secure Hash Standard. Federal Information Processing Standards Publication 180-4, U.S. Department of Commerce (March 2012), http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf
18. Nikolic, I., Biryukov, A.: Collisions for step-reduced SHA-256. In: Nyberg, K. (ed.) Fast Software Encryption – FSE 2008. LNCS, vol. 5086, pp. 1–15. Springer (2008)
19. Sanadhya, S.K., Sarkar, P.: New collision attacks against up to 24-step SHA-2. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) Progress in Cryptology – INDOCRYPT 2008. LNCS, vol. 5365, pp. 91–103. Springer (2008)
20. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer (2005)
21. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer (2005)
22. Yu, H., Bai, D.: Boomerang attack on step-reduced SHA-512. IACR Cryptology ePrint Archive, Report 2014/945 (2014), http://eprint.iacr.org/2014/945