Improving Local Collisions: New Attacks on Reduced SHA-256

Florian Mendel, Tomislav Nad, and Martin Schläffer

IAIK, Graz University of Technology, Austria
florian.mendel@iaik.tugraz.at

Abstract. In this paper, we focus on the construction of semi-free-start collisions for SHA-256, and show how to turn them into collisions. We present a collision attack on 28 steps of the hash function with practical complexity. Using a two-block approach we are able to turn a semi-free-start collision into a collision for 31 steps with a complexity of at most $2^{65.5}$. The main improvement of our work is to extend the size of the local collisions used in these attacks. To construct differential characteristics and confirming message pairs for longer local collisions, we had to improve the search strategy of our automated search tool. To test the limits of our techniques we present a semi-free-start collision for 38 steps.

Keywords: hash functions, SHA-2, cryptanalysis, collisions, semi-free-start collisions, differential characteristics, automatic search tool.

1 Introduction

Since 2005, many collision attacks have been shown for commonly used and standardized hash functions. In particular, the collision attacks of Wang et al. [17,18] on MD5 and SHA-1 have convinced many cryptographers that these widely deployed hash functions can no longer be considered secure. As a consequence, NIST has proposed the transition from SHA-1 to the SHA-2 family. Many companies and organization follow this advice and migrate to SHA-2. Additionally, SHA-2 is faster on many platforms than the recently chosen winner of the SHA-3 competition [14]. Hence, NIST explicitly recommends both, SHA-2 and SHA-3. Therefore, the cryptanalysis of SHA-2 is still of high interest.

In the last few years several cryptanalytic results have been published for SHA-256. The security of SHA-256 against preimage attacks was first studied by Isobe and Shibutani in [5]. They have presented a preimage attack on 24 out of 64 steps. This was improved by Aoki et al. to 43 steps in [1] and later extended to 45 steps by Khovratovich et al. in [6]. In [8] Li et al. have shown how a particular preimage attack can be used to construct a free-start collision attack for 52 steps of SHA-256. All attacks are only slightly faster than the generic attack which has a complexity of 2^{256} for preimages and 2^{128} for free-start collisions. Furthermore, second-order differential collisions for SHA-256 up to 47 steps with practical complexity have been shown in [2, 7].

T. Johansson and P. Nguyen (Eds.): EUROCRYPT 2013, LNCS 7881, pp. 262–278, 2013.
© International Association for Cryptologic Research 2013

In [15] Nikolić and Biryukov, studied the security of SHA-256 with respect to collision attacks. They found a differential characteristic resulting in a collision attack for 23 steps of SHA-256. Later this approach was extended to a collision attack on 24 steps [4, 16]. All these results use rather simple local collisions spanning over 9 steps, which are constructed mostly manually or using basic cryptanalytic tools. However, as pointed out in [4] it is unlikely that this approach can be extended beyond 24 steps.

Recently, Mendel et al. proposed a technique to use local collisions spanning over a higher number of steps. The main improvement is to use an automated search tool to construct the differential characteristics and to find confirming message pairs. Using local collisions spanning over more than 9 steps, a collision attack on 27 steps and a semi-free-start collision attack on 32 steps of SHA-256 has been shown. Both attacks have practical complexity. Currently, these are the best collision attacks on SHA-256 with practical complexity.

In this paper, we improve upon these collision attacks on SHA-256. We present collisions for the hash function on up to 31 out of 64 steps with complexity of at most $2^{65.5}$, and semi-free-start collisions on 38 steps with complexity of 2^{37}. We get these attacks by extending the size of the local collision to up to 18 steps. Furthermore, we try to ensure that the first message words do not contain any differences. This way, we can convert most of our semi-free-start collision attacks into collision attacks on the hash function.

The remainder of this paper is structured as follows. A description of the hash function is given in Sect. 2. A high-level overview of our attacks is given in Sect. 3. In Sect. 4 we show how we construct local collisions spanning over a higher number of steps. We show how to construct the differential characteristics in Sect. 5. In Sect. 6 we present our results and show how to turn (some of) the semi-free-start collision into collisions. Finally, we conclude in Sect. 7.

2 Description of SHA-256

SHA-256 is an iterated hash function that pads and processes the input message using t 512-bit message blocks m_j. The 256-bit hash value is computed using the compression function f:

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

In the following, we briefly describe the compression function f of SHA-256. It basically consists of two parts: the message expansion and the state update transformation. A more detailed description of SHA-256 is given in [13].

2.1 Message Expansion

The message expansion of SHA-256 splits the 512-bit message block into 16 words M_i, $i = 0, \ldots, 15$, and expands them into 64 expanded message words W_i as follows:

$$W_i = \begin{cases} M_i & 0 \le i < 16 \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} & 16 \le i < 64 \end{cases}$$

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

$$\sigma_0(x) = (x \ggg 7) \oplus (x \ggg 18) \oplus (x \gg 3)$$
$$\sigma_1(x) = (x \ggg 17) \oplus (x \ggg 19) \oplus (x \gg 10)$$

2.2 State Update Transformation

We use the alternative description of the SHA-256 state update given in [11], which is illustrated in Fig. 1.

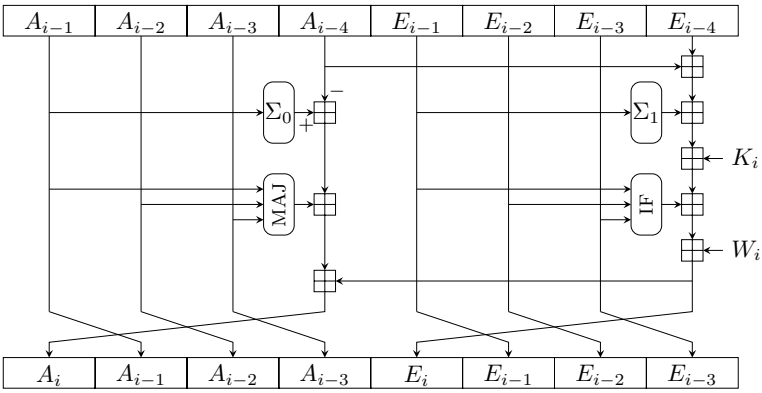


Fig. 1. The state update transformation of SHA-256

The state update transformation starts from the previous 256-bit chaining value $h_j = (A_{-4}, \ldots, A_{-1}, E_{-4}, \ldots, E_{-1})$ which is updated by applying the step functions 64 times. In each step $i = 0, \ldots, 63$ the expanded 32-bit word W_i is used to compute the two state variables E_i and A_i as follows:

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

For the definition of the step constants K_i we refer to [13]. The bitwise Boolean functions IF and MAJ used in each step are defined by

$$\mathrm{IF}(x, y, z) = xy \oplus xz \oplus z$$
$$\mathrm{MAJ}(x, y, z) = xy \oplus yz \oplus xz,$$

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

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

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 is the chaining value h_{j+1} for the next message block m_{j+1}.

3 Basic Attack Strategy

In this section, we give a high-level overview of our collision attacks on SHA-256. We first construct semi-free-start collisions for SHA-256 based on a local collision and then turn these collisions on the compression function into collisions on the hash function. This is possible if enough message words can be chosen freely at the input of the compression function. The main difficulty lies in the construction of semi-free-start collisions which offer enough free message words to turn them into collisions.

3.1 Constructing Local Collisions

If the local collision starts at step n, we get n free message words at the beginning of a differential characteristic. In this case message words $W_0, \ldots, W_{n-1}$ do not contain any differences. Hence, they can be chosen (almost) freely to match the initial value or the chaining value of the first block [4]. Note that in some cases, a few bits of these message words may be needed to fulfill conditions in the message expansion.

For our collision attack on 28 steps, we need a local collision with $t = 11$ steps which starts in step $n = 8$ and ends in step 18 (see Fig. 2). For our attack on 31 steps, we use a local collision with $t = 14$ steps which starts in step $n = 5$ and ends in step 18 (see Fig. 2). Although, these local collision spans over fewer words in E_i ($t - 4$ words) and A_i ($t - 8$ words), a lot of freedom is still needed to fulfill all the conditions imposed by the differential characteristic.

Therefore, we use local collisions which are sparse, especially in steps greater than 16. Since the message difference has the largest influence, we aim for a small number of message words which contain differences (see Sect. 4). Additionally, we try to keep the number of differences in the later words of A_i and E_i low. This significantly reduces the search space and improves the running time of our automatic search algorithm (see Sect. 5).

3.2 Turning Semi-Free-Start Collisions into Collisions

In SHA-256 a semi-free-start collision can easily be turned into a collision, if the first 8 message words can be chosen freely. In this case, we can choose the 8 message words to match the 8 words of the initial value [4]. We use this approach

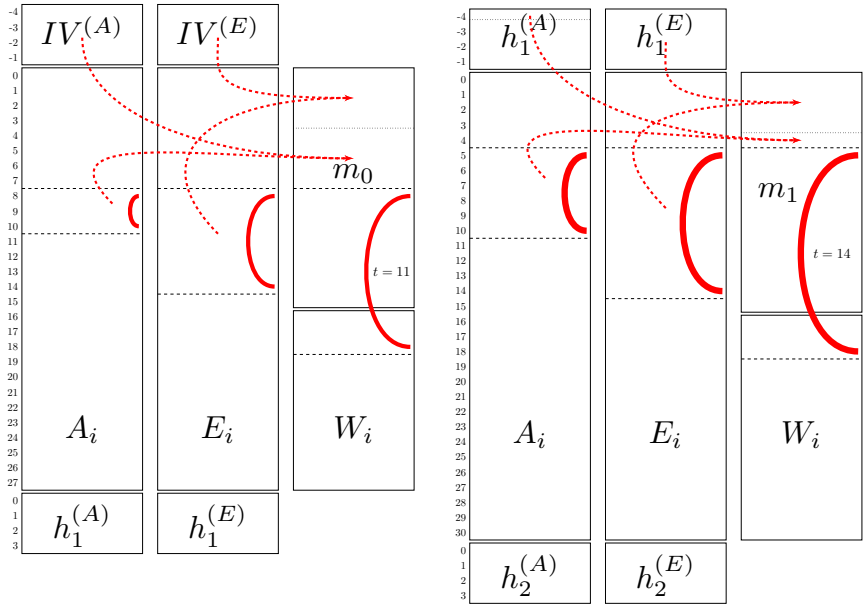


Fig. 2. Two approaches to construct collisions for SHA-256. The left approach uses 8 free message words to turn a semi-free-start collision on 28 steps into a collision on the hash function. The right approach uses random first blocks to increase the freedom and thus, the size of the local collision.

in our collision attack on 28 steps of SHA-256 (see Fig. 2). However, by attacking a higher number of steps a lot of freedom is needed for message modification, in particular after step 15. Therefore, we aim for local collisions resulting in only a few conditions after step 15, which limits the size of the local collision.

However, to get an attack on a higher number of steps, we need to increase the size of the local collision. This is possible by using a 2-block approach. In this case, we need less free message words at the beginning of the compression function. More specifically, by using an unbalanced meet-in-the-middle approach and computing 2^k random first blocks, we only need $256 - k$ free message bits to match the previous chaining value. If the complexity of finding a confirming message pair for the semi-free-start collision is 2^x, the total complexity of this approach will be $\max(2^k, 2^x \cdot 2^{128-k})$. We use this approach in our attack on 31 steps of SHA-256 (see Fig. 2). We construct the local collision such that the first 5 message words can be chosen freely.

3.3 Searching for Differential Characteristics and Message Pairs

A differential attack usually consists of two main steps: constructing a differential characteristic and finding a confirming message pair. Unfortunately, both steps are difficult and depend highly on the hash function under attack. For

our collision attacks on SHA-2 we proceed according to the following high-level overview:

- **Find a differential characteristic**
  1. Choose (sparse) message difference
  2. Determine sparse part of the differential characteristic
  3. Determine dense part of the differential characteristic
- **Find a confirming message pair**
  4. Use message modification in dense part of the characteristic
  5. Perform random trials in sparse part of the characteristic
  6. Use free message words to match the initial value (optional)

To search for differential characteristics and confirming message pairs, we use the same approach and automatic search tool as in [11]. However, the selection of starting points for the search (message words which contain differences) and the search strategy have been improved. We try to minimize both, the number of message words which contain differences and the conditions on the expanded message words (see Sect. 4). This allows us to construct sparser differential characteristics and increases the freedom in the message words.

We start the search by first constructing a differential characteristic for the message expansion. Next, we guess on the last few state words of the local collision to keep the differential characteristic sparse towards the end. This improved guessing strategy is essential for our attack to work. Finally, we continue the search to find a confirming message pair.

4 Determining Message Words with Differences

In this section, we show how to construct local collisions, which result in a (semi-free-start) collision attack on a high number of rounds. The previously best results on SHA-256 have been published by Mendel et al. in [11]. Using a local collision on $t = 11$ steps with differences in 5 message words they have shown a collision for 27 steps of the SHA-256 hash function with practical complexity. Additionally, a local collision on $t = 16$ steps with differences in 8 expanded message words has been used to construct semi-free-start collisions on 32 steps of SHA-256. To improve on these attacks, we need to increase the length t of the local collision, while still keeping the part without differences large.

4.1 Constructing Local Collisions in SHA-256

Compared to previous attacks [4,11,15,16], we are able to find differential characteristics over a much larger number of steps in the state update of SHA-256 (see Sect. 5). This allows us to increase the size t of the local collision. However, for large values of t the complexity of a local collision increases significantly. The larger the value of t, the more freedom is needed to find a confirming message pair. Since we additionally need free message words to turn a semi-free-start collision into a collision, we need to reduce the complexity of the local collision.

Table 1. Message word differences and message word conditions for the attacks on 28 steps (left), 31 steps (middle) and 38 steps (right) steps of SHA-256. Rows show the individual steps of the message expansion to compute W_i. Columns (and highlighted rows) show those expanded message words which contain a difference. An occurrence of a message word in the message expansion equation is denoted by 'x'. For all rows which are not highlighted but contain an 'x', the message differences must cancel.

W_i	8	9	13	16	18
0					
1					
2					
3					
4					
5					
6					
7					
8	x				
9		x			
10					
11					
12					
13			x		
14					
15					
16		x		x	
17					
18				x	x
19					
20			x		x
21					
22					
23	x			x	
24	x	x			
25		x			x
26					
27					

W_i	5	6	7	8	9	16	18
0							
1							
2							
3							
4							
5	x						
6		x					
7			x				
8				x			
9					x		
10							
11							
12							
13							
14							
15							
16					x	x	
17							
18						x	x
19							
20	x						x
21	x	x					
22		x	x				
23			x	x		x	
24				x	x		
25					x		x
26							
27							
28							
29							
30							

W_i	7	8	10	15	23	24
0						
1						
2						
3						
4						
5						
6						
7	x					
8		x				
9						
10			x			
11						
12						
13						
14						
15				x		
16						
17			x	x		
18						
19						
20						
21						
22	x			x		
23	x	x			x	
24		x				x
25			x		x	
26			x			x
27						
28						
29						
30				x	x	
31				x		x
32						
33						
34						
35						
36						
37						

This is possible be reducing the number of message words which contain differences. The freedom of every message word which does not contain a difference can be used more easily for message modification. Additionally, we require that the expanded message words cancel each other, such that we get a high number of steps which do not contain any differences at all. This cancellation of message word differences results in conditions on the message expansion, and fewer message words with differences result in fewer conditions.

To find good candidates for local collisions and message word differences, we first need to determine the initial constraints. To turn a semi-free-start collision into a collision attack, we need to have no differences in the first (ideally 8) message words. Furthermore, we consider only local collisions with $t > 9$ which result in (semi-free-start) collision attacks on more than 27 steps. We prefer local collisions with small values of t and which result in fewer conditions on the expanded message words. In particular, we try to avoid many conditions (and thus, message words with differences) after step 16 since they are more difficult to fulfill.

4.2 Candidates of Good Local Collisions

For each candidate of t, we identify those message words which must have differences such that the differential characteristic holds for a large number of steps. Then, we minimize the number of conditions on the expanded message and the number of message words with differences. Using this strategy, we get many possible candidates for good local collisions. For the three candidates shown in Table 1 we are able to construct differential characteristics (see Sect. 5) and present collision and semi-free-start collision attacks (see Sect. 6).

The first local collision in Table 1 spans over $t = 11$ steps (step 8-18) and results in a collision on 28 steps of SHA-256. The local collision has differences in only 5 message words $(W_8, W_9, W_{13}, W_{16}, W_{18})$ and we get 4 conditions in step 20, 23, 24, and 25 of the message expansion. The first 8 message words do not contain any differences and therefore, we can turn a semi-free-start collision into a real collision for SHA-256.

To extend the collision attacks on SHA-256 to more steps, we drop the condition that the first 8 message words should have no differences. If only the 5 first message words contain no differences, we can still get an attack below the birthday bound using a 2-block approach (see Sect. 3). We get a local collision spanning over $t = 14$ steps (step 5-18) with differences in only 7 expanded message words $(W_5, \ldots, W_9, W_{16}, W_{18})$ which results in a collision attack on 31 steps of SHA-256. We get 6 conditions in step 20–25 of the message expansion (see Table 1).

Finally, if we only search for local collisions which result in a semi-free-start collision, we can extend the number of steps to attack even further. Using a local collision spanning over $t = 18$ steps with differences in only 6 expanded message words $(W_7, W_8, W_{10}, W_{15}, W_{23}, W_{24})$ we can get an attack on up to 38 steps of SHA-256. We get conditions in 6 steps of the message expansion (see Table 1).

5 Finding Differential Characteristics

After the message words which contain differences are fixed, we need to find a differential characteristic. Due to the increased complexity of SHA-2 compared to other members of the MD4 family, finding good (high probability) differential characteristics by hand is almost impossible. We use the techniques developed by Mendel et al. which have been applied in several attacks on ARX-based hash functions [9–12]. Using an automated search tool, complex nonlinear differential characteristics can be found. Additionally, the tool can be used for solving nonlinear equations involving conditions on state and message words (i.e. for message modification).

5.1 Automated Search for Differential Characteristics

The basic idea of the search algorithm is to pick and guess previously unrestricted bits. After each guess, the information due to these restrictions is propagated to other bits. If an inconsistency occurs, the algorithm backtracks to an earlier state of the search and tries to correct it. Similar to [11], we denote these three parts of the search by decision (guessing), deduction (propagation), and backtracking (correction). Then, the search algorithm proceeds as follows:

Let U be a set of bits. Repeat the following until U is empty:

Decision (Guessing)

1. Pick randomly (or according to some heuristic) a bit in U.
2. Impose new constraints on this bit.

Deduction (Propagation)

3. Propagate the new information to other variables and equations as described in [11].
4. If an inconsistency is detected start backtracking, else continue with step 1.

Backtracking (Correction)

5. Try a different choice for the decision bit.
6. If all choices result in an inconsistency, mark the bit as critical.
7. Jump back until the critical bit can be resolved.
8. Continue with step 1.

In the deduction, we use generalized conditions on bits [3] to propagate information. A generalized condition takes all 16 possible conditions on a pair of bits into account. Table 2 lists these conditions and introduces a notation for them. We also use generalized conditions to represent the differential characteristics in the remainder of this paper. During the search, we propagate information and backtrack as proposed in [11]. Similar to [11], we additionally consider linear conditions on two related bits ($X_j \oplus X_k = \{0, 1\}$) during the search for a confirming message pair.

The main difficulty in finding a long differential characteristic lies in the fine-tuning of the search algorithm. There are a lot of variations possible which can

Table 2. Notation for all generalized conditions on a pair of bits [3]

(X_i, X_i^*)	(0,0)	(1,0)	(0,1)	(1,1)	(X_i, X_i^*)	(0,0)	(1,0)	(0,1)	(1,1)
?	✓	✓	✓	✓	3	✓	✓	-	-
-	✓	-	-	✓	5	✓	-	✓	-
x	-	✓	✓	-	7	✓	✓	✓	-
0	✓	-	-	-	A	-	✓	-	✓
u	-	✓	-	-	B	✓	✓	-	✓
n	-	-	✓	-	C	-	-	✓	✓
1	-	-	-	✓	D	✓	-	✓	✓
#	-	-	-	-	E	-	✓	✓	✓

decide whether the search succeeds or fails. The main improvement compared to [11] results from an improved decision (guessing) part of the search algorithm. Instead of randomly choosing bits from the whole set of unrestricted bits, we split the set in specific sub-sets. These sub-sets are chosen such that the resulting differential characteristic gets sparser and the search terminates faster. We cover these improvements which led to the new results on SHA-256 in the following section.

5.2 Improved Decision Strategy for SHA-256

Note that already the starting point has a large influence on the efficiency of finding a differential characteristic. In the case of our attacks on SHA-256, we have chosen a starting point where the local collision is not so long and only a few message words contain differences. This significantly reduces the search space for the automatic search tool. To further improve the efficiency of the tool, we reduce the search space further by separating the search into 3 stages, compared to two stages in [11].

In each stage, we define a different set of unrestricted bits in U_i. This way, we can control the order in which we guess bits. For the local collisions given in the previous section, the best strategy is to first search for a differential characteristic in the message expansion. Then, we continue by searching for a sparse characteristic in the state update and finally, we search for a confirming message pair. The stages are executed sequentially but we dynamically switch between them if a contradiction occurs. If necessary, we try to correct contradictions by backtracking into the previous stages and continue the search from there. Additionally, we restart the search from scratch after a certain amount of inconsistencies occurred. This terminates search branches for which it is unlikely that a solution can be found. The three stages can be summarized as follows:

Stage 1: We first search for a consistent differential characteristic in the message expansion. Hence, we only add unconstrained bits '?' or 'x' of W_i to the set U_1. Furthermore, we try to reduce the number of conditions after step 15 in the message expansion. In this case, it is more likely to find confirming message pairs in the last stage of the search. To get a sparser characteristic

in this area, we pick decision bits more often from the last few steps of the message expansion.

Stage 2: Once we have found a differential characteristic for the message expansion, we continue with searching for a differential characteristic in the state update. We add all unconstrained bits '?' or 'x' of A and E to the set U_2. Note that we pick decision bits more often from A, since this results in sparser characteristics for A. Similar to Stage 1, experiments have shown that in this case, confirming message pairs are easier to find in the last stage.

Stage 3: In the last stage, we search for confirming inputs. We only pick decision bits '-' which are constraint by linear two-bit conditions, similar as in [11]. This ensures that those bits which influence a lot of other bits are guessed first. Additionally, at least all bits influenced by two-bit conditions propagate as well. This way, inconsistent characteristics can be detected earlier and valid solutions are found faster.

Note that after Stage 3 finishes, we already get a confirming message pair which results in a semi-free-start collision. The corresponding differential characteristics for 28, 31 and 38 steps of SHA-256, including bits marked with linear conditions on two bits are given in Table 3, 4 and 5. Note that for SHA-2, the characteristics are in general too complex to list all conditions (including non-linear conditions on two or more bits). Therefore, all our characteristics are verified by providing conforming message pairs in the appendix.

6 Finding Confirming Message Pairs

In this section, we present our results and show how to turn (some of) the semi-free-start collision into collisions on the hash function. To confirm our claims, we present confirming message pairs for those steps of our attacks which have practical complexity.

6.1 Collision for 28 Steps of SHA-256

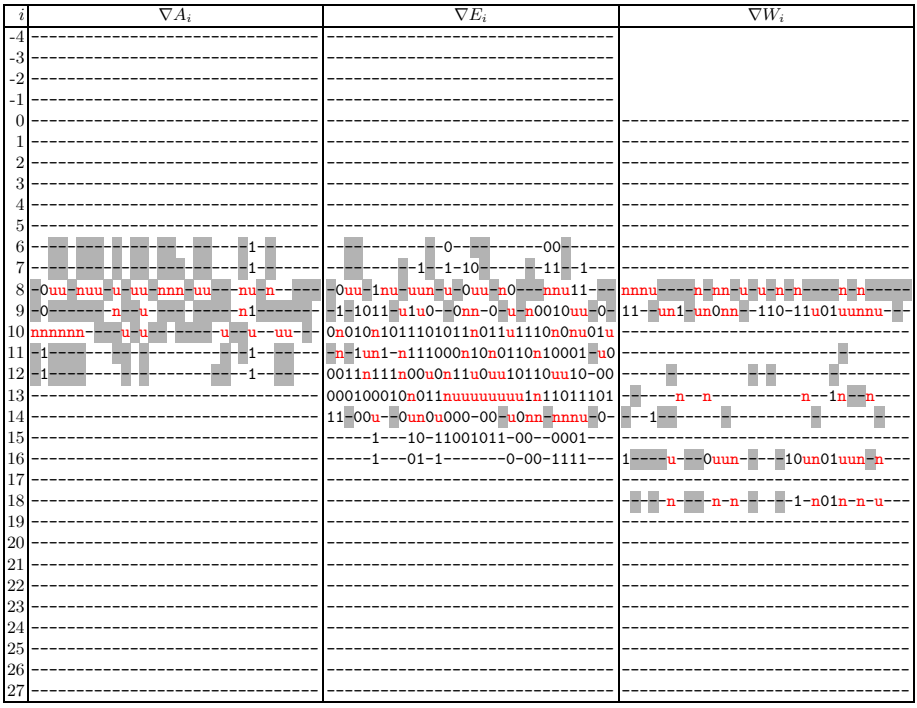
Using the starting point given in Table 1 and the search strategy described in Sect. 5, we can find a semi-free-start collision for 28 steps of SHA-256. Finding the differential characteristic took about 5 hours on a single cpu, which is equivalent to about $2^{36.3}$ SHA-256 evaluations using the current version of OpenSSL as a reference point.

Since we can (almost) freely choose the first 8 message words, this semi-free-start collision can be turned into a collision on the hash function with almost no cost. The confirming message pair is given in Table 6 and the according differential characteristic is given in Table 3.

6.2 Collision for 31 Steps of SHA-256

Using the starting point given in Table 1, we can find a semi-free-start collision on 31 steps of SHA-256, where the first 5 message words can be chosen freely.

Table 3. Differential characteristic for the collision attack on SHA-256 reduced to 28 steps. Bits with gray background have at least one additional condition.

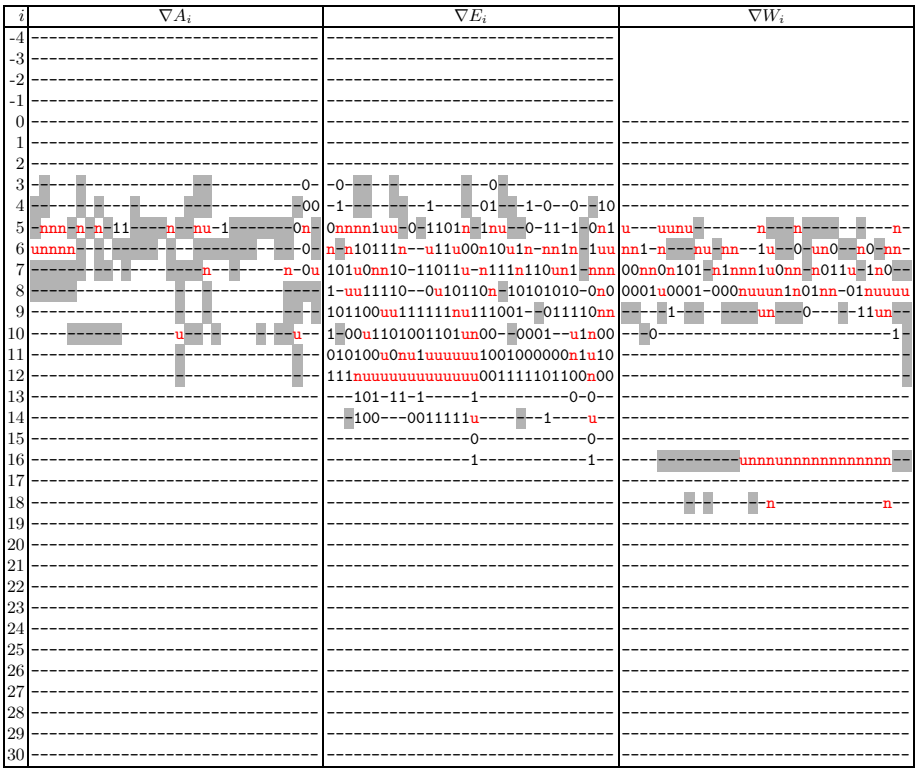


Finding the differential characteristic and confirming message pair for the semi-free-start collision took about 21 hours on a single CPU, which is equivalent to about $2^{38.4}$ SHA-256 evaluations. The resulting differential characteristic is given in Table 4. To demonstrate that we can indeed choose the 5 first message words, we have set the last 5 chaining words to 0 (see Table 7).

In the characteristic on 31 steps, we have no differences in the first 5 message words. Using a single semi-free-start collision and a two-block approach, we can construct a collision for the SHA-256 hash function reduced to 31 steps with a complexity of about $2^{99.5}$ compression function evaluations. We start this part of the attack with the differential characteristic given in Table 4 and continue as follows:

1. Use the automatic tool to determine all expanded message words and state variables in steps 5–12. This also determines the state words E_1–E_4 and A_{-3}–A_4. Note that this step of the attack takes only seconds and does not contribute to the final attack complexity.
2. Compute 2^{96} arbitrary first message blocks to fulfill the 96 conditions on the chaining input A_{-3}–A_{-1}. This step of the attack has a complexity of about 2^{96} SHA-256 evaluations and also determines the expanded message words W_0–W_4.

Table 4. Differential characteristic for the collision attack on SHA-256 reduced to 31 steps. Bits with gray background have at least one additional condition.



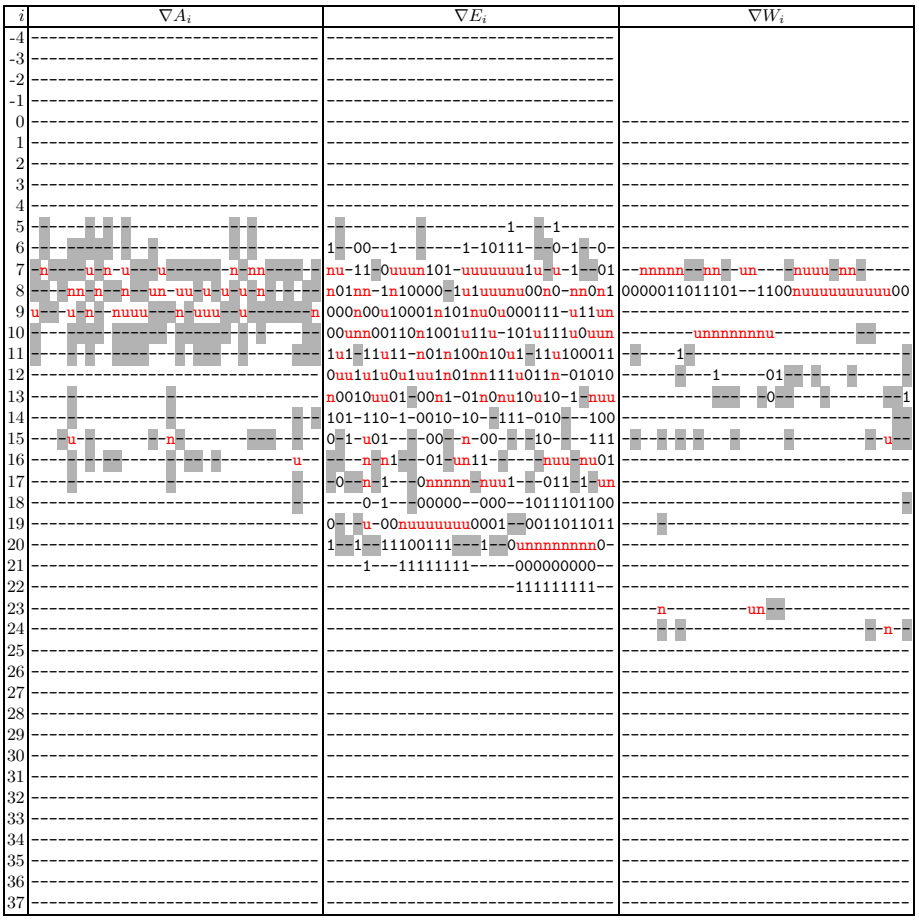
3. At this point, the chaining value and the message words W_0–W_{12} are chosen. Next, we use the freedom in the message words W_{13}–W_{15} to fulfill the conditions on E_{13}–E_{15} and W_{16}, W_{18}. However, this step of the attack succeeds only with a probability of about 1/12 (verified experimentally) since we have just not enough freedom in W_{13}–W_{15}. If this step fails then we go back to step 2.

To summarize, we can find a 2-block collision for SHA-256 reduced to 31 steps with a complexity of about $12 \cdot 2^{96} \approx 2^{99.5}$ SHA-256 evaluations. Note that the complexity of the attack can be improved significantly by using a meet-in-the-middle approach.

Instead of computing only one solution in the first step of the attack, we compute ℓ solutions and save them in a list L. In step 2 of the attack where we compute an arbitrary first message blocks, we check for a match in the list L. By increasing the size ℓ of the list, we can reduce the number of first message blocks that we need to compute by a factor of $1/\ell$.

The main question is how many entries in L can be computed in our attack. Using our unoptimized code we have already found $\ell > 2^{19.5}$ different solutions for step 1 of the attack with an average cost of $2^{25.5}$. This reduces the attack

Table 5. Differential characteristic for the semi-free-start collision attack on SHA-256 reduced to 38 steps. Bits with gray background have at least one additional condition.



complexity to about 2^{80} SHA-256 evaluations. However, our experiments indicate that we can expect to find about 2^{34} different solutions, which results in a total attack complexity of about $2^{65.5}$ SHA-256 evaluations.

6.3 Semi-Free-Start Collision for 38 Steps of SHA-256

Finally, we have also improved the best semi-free-start collision attack on SHA-256. Using the starting point given in Table 1, we can find a semi-free-start collision for 38 steps of SHA-256. Finding the differential characteristic and the confirming message pair took about 8 hours on a single CPU. This is equivalent to about 2^{37} SHA-256 evaluations. The differential characteristic is shown in Table 5 and the resulting semi-free-start collision in Table 8.

7 Conclusions

In this paper, we have improved the best known collision attacks on the SHA-256 hash function from 27 to 31 steps. We focus on the construction of semi-free-start collisions which can be turned into collisions on the hash function. Our results are hash function collisions on 31 steps of SHA-256 with complexity $2^{65.5}$, and semi-free-start collisions on 38 steps with complexity 2^{37}. We have verified all our attacks by providing practical examples whenever this was possible.

Our results were obtained by extending the size of the local collision up to 18 steps. Furthermore, we ensure that the first message words do not contain any differences and try to reduce the number of conditions on the expanded message words. To find differential characteristics and confirming message pairs for local collisions spanning over more steps we have improved the efficiency of the automatic search tool used by Mendel et al. in their attacks on reduced SHA-256 in several ways. Most importantly, we have improved the search strategy to find sparser differential characteristics by guessing primarily bits towards the end of the local collision.

Acknowledgments. Part of this work was done while Florian Mendel was with KU Leuven. The work has been supported in part by the Secure Information Tech- nology Center-Austria (A-SIT) and by the Austrian Science Fund (FWF), project P21936-N23.

References

1. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for Step-Reduced SHA-2. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 578–597. Springer, Heidelberg (2009)
2. Biryukov, A., Lamberger, M., Mendel, F., Nikolić, I.: Second-Order Differential Collisions for Reduced SHA-256. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 270–287. Springer, Heidelberg (2011)
3. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, Heidelberg (2006)
4. 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.) SAC 2008. LNCS, vol. 5381, pp. 276–293. Springer, Heidelberg (2009)
5. Isobe, T., Shibutani, K.: Preimage Attacks on Reduced Tiger and SHA-2. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 139–155. Springer, Heidelberg (2009)
6. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 244–263. Springer, Heidelberg (2012)
7. Lamberger, M., Mendel, F.: Higher-Order Differential Attack on Reduced SHA-256. Cryptology ePrint Archive, Report 2011/037 (2011), `http://eprint.iacr.org/`
8. 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.) FSE 2012. LNCS, vol. 7549, pp. 264–286. Springer, Heidelberg (2012)

9. Mendel, F., Nad, T., Scherz, S., Schläffer, M.: Differential Attacks on Reduced RIPEMD-160. In: Gollmann, D., Freiling, F.C. (eds.) ISC 2012. LNCS, vol. 7483, pp. 23–38. Springer, Heidelberg (2012)
10. Mendel, F., Nad, T., Schläffer, M.: Cryptanalysis of Round-Reduced HAS-160. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 33–47. Springer, Heidelberg (2012)
11. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 288–307. Springer, Heidelberg (2011)
12. Mendel, F., Nad, T., Schläffer, M.: Collision Attacks on the Reduced Dual-Stream Hash Function RIPEMD-128. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 226–243. Springer, Heidelberg (2012)
13. National Institute of Standards and Technology: FIPS PUB 180-3: Secure Hash Standard. Federal Information Processing Standards Publication 180-3, U.S. Department of Commerce (October 2008), `http://www.itl.nist.gov/fipspubs`
14. National Institute of Standards and Technology: SHA-3 Selection Announcement (October 2012), `http://csrc.nist.gov/groups/ST/hash/sha-3/sha-3_selection_announcement.pdf`
15. Nikolić, I., Biryukov, A.: Collisions for Step-Reduced SHA-256. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 1–15. Springer, Heidelberg (2008)
16. Sanadhya, S.K., Sarkar, P.: New Collision Attacks against Up to 24-Step SHA-2. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 91–103. Springer, Heidelberg (2008)
17. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
18. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

A Resulting Message Pair Examples

Table 6. Example of a collision for 28 steps of SHA-256

h_0	6a09e667 bb67ae85 3c6ef372 a54ff53a 510e527f 9b05688c 1f83d9ab 5be0cd19
m	14c48440 b3c3277f ad69812d c3d4dffa 7eae690b 7f9fe027 832aece8 9a489458
	1607a45c db81bdc8 8786e031 d8f22801 72b6be5e 45a2652f f3fbb17a 2ce70f52
m^*	14c48440 b3c3277f ad69812d c3d4dffa 7eae690b 7f9fe027 832aece8 9a489458
	e6b2f4fc d759b930 8786e031 d8f22801 72b6be5e 47e26dbf f3fbb17a 2ce70f52
Δm	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
	f0b550a0 0cd804f8 00000000 00000000 00000000 02400890 00000000 00000000
h_1	01470131 cd0062bc 7e8f8c21 98938652 3d49075a 327f38e8 11f0d36d 58601725

Table 7. Example of a semi-free-start collision for 31 steps of SHA-256 with the last 5 chaining variables set to 0

h_0	532f13f5 6a28c3c0 e301fab5 00000000 00000000 00000000 00000000 00000000
m	d55c884f faf18f34 b772b323 af46235b 3d8bd87b dd3e8271 26618488 02d189d0
	1883a4af 4f99167b 271b11c7 81b8363d b27e389d 2155a533 8b811348 4a8da291
m^*	d55c884f faf18f34 b772b323 af46235b 3d8bd87b 523f9273 eeb902ae 36ff3d98
	108477b0 4f989677 271b11c7 81b8363d b27e389d 2155a533 8b811348 4a8da291
Δm	00000000 00000000 00000000 00000000 00000000 8f011002 c8d88626 342eb448
	0807d31f 0001800c 00000000 00000000 00000000 00000000 00000000 00000000
h_1	6ff5a9a7 9d014158 12938ebb dbf8dc76 29fb2c4c b48b053e 1c4377a9 e21554c1

Table 8. Example of a semi-free-start collision for 38 steps of SHA-256

h_0	ba75b4ac c3c9fd45 fce04f3a 6d620fdb 42559d01 b0a0cd10 729ca9bc b284a572
m	4f5267f8 8f8ec13b 22371c61 56836f2b 459501d1 8078899e 98947e61 4015ef31
	06e98ffc 4babda4a 27809447 3bf9f3be 7b3b74e1 065f711d 6c6ead5e a1781d54
m^*	4f5267f8 8f8ec13b 22371c61 56836f2b 459501d1 8078899e 98947e61 7e73f1f1
	06e99000 4babda4a 277f1447 3bf9f3be 7b3b74e1 065f711d 6c6ead5e a1781d50
Δm	00000000 00000000 00000000 00000000 00000000 00000000 00000000 3e661ec0
	00001ffc 00000000 00ff8000 00000000 00000000 00000000 00000000 00000004
h_1	baa8df17 9f9f64dd d57d5c2c 7b232c81 1f3916e6 7a03a2be 7afb1d86 6b0eced6