Higher-Order Differential Attack on Reduced SHA-256

Mario Lamberger and Florian Mendel

Institute for Applied Information Processing and Communications
Graz University of Technology, Inffeldgasse 16a, A–8010 Graz, Austria.
florian.mendel@iaik.tugtraz.at

Abstract. In this work, we study the application of higher-order differential attacks on hash functions. We show a second-order differential attack on the SHA-256 compression function reduced to 46 out of 64 steps. We implemented the attack and give the result in Table 1. The best attack so far (in a different attack model) with practical complexity was for 33 steps of the compression function.

1 Introduction

In recent years, significant advances in the field of hash function research have been made which had a formative influence on the landscape of hash functions. Especially the work on MD5 and SHA-1 has convinced many cryptographers that these widely deployed hash functions can no longer be considered secure [29,30]. As a consequence, people are evaluating alternative hash functions, *e.g.* in the SHA-3 initiative organized by NIST [24]. During this ongoing evaluation, not only the three classical security requirements are considered. Researchers look at (semi-) free-start collisions, near-collisions, etc. Whenever a 'behavior different from that expected of a random oracle' can be demonstrated for a new hash function, it is considered suspect, and so are weaknesses that are demonstrated only for the compression function and not for the full hash function.

With the cryptographic community joining forces in the SHA-3 competition, the SHA-2 family gets considerably less attention. Apart from being marked as 'relying on the same design principle as SHA-1 and MD5', the best attack to date on SHA-256 is a collision attack for 24 out of 64 steps with practical complexity [10,26] and a preimage attack on 43 steps [1] having a complexity of $2^{254.9}$.

In this work, we present an attack for the SHA-256 compression function reduced to 46 out of 64 steps with practical complexity. The attack is an application of higher-order differentials on hash functions. Table 1 shows the resulting example.

Higher-order differentials have been introduced by Lai in [14] and first applied to block ciphers by Knudsen in [13]. The application to stream ciphers was proposed by Dinur and Shamir in [9] and Vielhaber in [27]. First attempts to apply these strategies to hash functions were published in [2].

Table 1. Example of a second-order differential collision $f(y + a_1 + a_2) - f(y + a_1) - f(y + a_2) + f(y) = 0$ for 46 steps of the SHA-256 compression function. Values and differences are given in hexadecimal notation.

y	72939135 c1570fea 5c5d0c1d ad031d03
	d83c56b6 41334f38 12f67844 0edd1fcb
	016a5c6f 39094c7b 9e181d92 54bfb0fa
	506781eb 0b081e5e 607a28e0 6318673a
	21315086 43909ad8 23e8771b 26ca42e8
	1eecc4dd 14649b3d 9076304c 29f92e96
a_1	00000000 00000000 00000000 00000000
	00000000 00000000 fffffffc 00000000
	00000000 00000000 00000000 00000000
	00000000 00000000 00000000 00000000
	00000000 00000004 00000000 00000004
	ef800200 fffffe00 0ffffe00 efbef7fc
a_2	a3a3e47f 58cf0adb 3fa82cc6 0c907f06
	3377c2cd 1997456a a8bcf700 2455c931
	bffb159c 504e97b0 39e6d04a 2a582f18
	37fcb1a0 d42be48e 950d8d60 ed368ca3
	e4ae4c2e 989ee693 8dd81c5e 3abd607d
	96c9d3bd 589c4e77 1a4afee7 b9ba6518

Recently, higher-order differential attacks have been applied to several hash functions submitted to SHA-3 initiative organized by NIST such as Hamsi [7], Keccak [7], and Luffa [32]. All these hash functions have in common that they rely on a completely different design principle than SHA-256. All are permutation-based designs following the sponge construction [3].

2 Higher-Order Differential Attacks on Hash Functions

In this section, we give a high-level description of the attack. It is an application of higher-order differential cryptanalysis on hash functions. While a standard differential attack exploits the propagation of the difference between a pair of inputs to the corresponding output differences, a higher-order differential attack exploits the propagation of the difference between differences.

Higher-order differential cryptanalysis was introduced by Lai in [14] and subsequently applied by Knudsen in [13]. We recall the basic definitions that we will need in the subsequent sections.

Definition 1. *Let $(S, +)$ and $(T, +)$ be Abelian groups. For a function $f : S \to T$, the derivative at a point $a_1 \in S$ is defined as*

$$\Delta_{(a_1)} f(y) = f(y + a_1) - f(y)\,. \tag{1}$$

The i-th derivative of f at $(a_1, a_2, \ldots, a_i)$ is then recursively defined as

$$\Delta_{(a_1,\ldots,a_i)} f(y) = \Delta_{(a_i)}(\Delta_{(a_1,\ldots,a_{i-1})} f(y)) \,. \tag{2}$$

Definition 2. *A one round differential of order i for f is an $(i+1)$-tuple $(a_1, a_2, \ldots, a_i; b)$ such that*

$$\Delta_{(a_1,\ldots,a_i)} f(y) = b \,. \tag{3}$$

When applying differential cryptanalysis to a hash function, a collision for the hash function corresponds to a pair of inputs with output difference zero. Similarly, when using higher-order differentials we define a higher-order differential collision for a function as follows.

Definition 3. *An i-th-order differential collision for a function f is an i-tuple $(a_1, a_2, \ldots, a_i)$ together with a value y such that*

$$\Delta_{(a_1,\ldots,a_i)} f(y) = 0 \,. \tag{4}$$

Note that the common definition of a collision for hash functions corresponds to a higher-order differential collision of order $i = 1$.

Now we want to talk about the *query complexity* of a differential collision of order i for a function f having an n-bit output. We are only allowed oracle access to f and ignore all other computations, memory accesses, etc.

From (4) we see that we can freely choose $i + 1$ of the input parameters which then fix the remaining ones. Hence, the expected number of solutions to (4) is one after choosing $2^{n/(i+1)}$ values for each of y and $a_1, \ldots, a_i$ so the query complexity of a differential collision of order i, for a good hash/compression function f is:

$$\approx (i+1) \cdot 2^{n/(i+1)} \tag{5}$$

We want to note that the complexity might be much higher in practice than this bound for the query complexity. In the following, we will show how to construct a second-order differential collision for the compression function of a block-cipher-based hash function. We are not aware of any algorithm for the case $i = 2$ faster than $2^{n/2}$.

2.1 Higher-Order Differential Collision for Block-Cipher-Based Compression Functions

In all of the following, we consider block ciphers $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ where n denotes the block length and k is the key length. For our purposes, we will also need to endow $\{0,1\}^n$ with an additive group operation. It is however not important, in which way this is done. A natural way would be to simply use the XOR operation on $\{0,1\}^n$ or the identification $\{0,1\}^n \leftrightarrow \mathbb{Z}_{2^n}$ and define the addition of $a, b \in \{0,1\}^n$ by $a + b \bmod 2^n$. Alternatively, if we have an integer

w dividing n, that is $n = \ell \cdot w$, we can use the bijection of $\{0,1\}^n$ and $\mathbb{Z}_{2^w}^\ell$ and define the addition as the word-wise modular addition, that is,

$$(\{0,1\}^n, +) := \underbrace{(\mathbb{Z}_{2^w}, +) \times \cdots \times (\mathbb{Z}_{2^w}, +)}_{\ell \text{ times}}. \tag{6}$$

The latter definition clearly aims very specifically at the SHA-2 design, *cf.* Section 3. However, the particular choice of the group law has no influence on our attack.

A well known construction to turn a block cipher into a compression function is the Davies-Meyer construction. The compression function call to produce the i-th chaining value x_i from the i-th message block and the previous chaining value x_{i-1} has the form:

$$x_i = E(m_i, x_{i-1}) + x_{i-1} \tag{7}$$

When attacking block-cipher-based hash functions, the key is *not* a secret parameter so for the sake of readability, we will slightly restate the compression function computation (7) where we consider an input variable $y = (k||x) \in \{0,1\}^{k+n}$ so that a call to the block cipher can be written as $E(y)$. Then, the Davis-Meyer compression function looks like:

$$f(y) = E(y) + \tau_n(y), \tag{8}$$

where $\tau_n(y)$ represents the n least significant bits of y.

In an analogous manner, we can also write down the compression functions for the Matyas-Meyer-Oseas and the Miyaguchi-Preneel mode which are all covered by the following proposition.

Proposition 1 *For any block-cipher-based compression function which can be written in the form*

$$f(y) = E(y) + L(y), \tag{9}$$

where L is a linear function with respect to $+$, an i-th-order differential collision for the block cipher transfers to an i-th-order collision for the compression function for $i \geq 2$.

For the proof of Proposition 1, we will need following property of $\Delta_{(a_1,\ldots,a_i)} f(y)$:

Proposition 2 (Lai [14]) *If $\deg(f)$ denotes the non-linear degree of a multivariate polynomial function f, then*

$$\deg(\Delta_{(a)} f(y)) \leq \deg(f(y)) - 1\,. \tag{10}$$

Proof (of Proposition 1). Let $\Delta_{(a_1,\ldots,a_i)} E(y) = 0$ be an i-th-order differential collision for $E(y)$. Both the higher-order differential and the mode of operation for the compression function are defined with respect to the same additive operation on $\{0,1\}^n$. Thus, from (9) we get

$$\Delta_{(a_1,\ldots,a_i)}(E(y) + L(y)) = \Delta_{(a_1,\ldots,a_i)} E(y) + \Delta_{(a_1,\ldots,a_i)} L(y),$$

so we see that all the terms vanish because the linear function $L(y)$ has degree one and so for $i \geq 2$ we end up with an i-th-order differential collision for the compression function because of Proposition 2. ■

2.2 Second-Order Differential Collision for the Block-Cipher-Based Compression Functions

The main idea of the attack is now to use two independent high probability differential characteristics – one in forward and one in backward direction – to construct a second-order differential collision for the block cipher E and hence due to Proposition 1 for the compression function.

Therefore, the underlying block cipher E is split into two subparts, $E = E_1 \circ E_0$. Furthermore, assume we are given two first-order differentials for the two subparts, where one holds in the forward direction and one in the backward direction and we assume that both have high probability. This part of the strategy has been already applied in other cryptanalytic attacks, we refer to Section 2.3 for related work. We also want to stress, that due to our definition above, the following differentials are actually related-key differentials. We have

$$E_0^{-1}(y + \beta) - E_0^{-1}(y) = \alpha \tag{11}$$

and

$$E_1(y + \gamma) - E_1(y) = \delta \tag{12}$$

where the differential in E_0^{-1} holds with probability p_0 and in E_1 holds with probability p_1. Using these two first-order differentials, we can now construct a second-order differential collision for the block cipher E. This can be summarized as follows (see also Figure 1).

1. Choose a random value for X and compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
2. Compute backward from X, X^*, Y, Y^* using E_0^{-1} to obtain P, P^*, Q, Q^*.
3. Compute forward from X, X^*, Y, Y^* using E_1 to obtain C, C^*, D, D^*.
4. Check if $P^* - P = Q^* - Q$ and $D - C = D^* - C^*$ is fulfilled.

Due to (11) and (12),

$$P^* - P = Q^* - Q = \alpha, \text{ resp. } D - C = D^* - C^* = \delta, \tag{13}$$

will hold with probability at least p_0^2 in the backward direction, resp. p_1^2 in the forward direction. Hence, assuming that the differentials are independent the attack succeeds with a probability of $p_0^2 \cdot p_1^2$. It has to be noted that this independence assumption is quite strong, *cf.* [22]. However, if this assumption holds, the expected number of solutions to (13) is 1, if we repeat the attack about $1/(p_0^2 \cdot p_1^2)$ times. As mentioned before, in our case, there is no secret key involved, so message modification techniques (*cf.* [30]) can be used to improve this complexity.

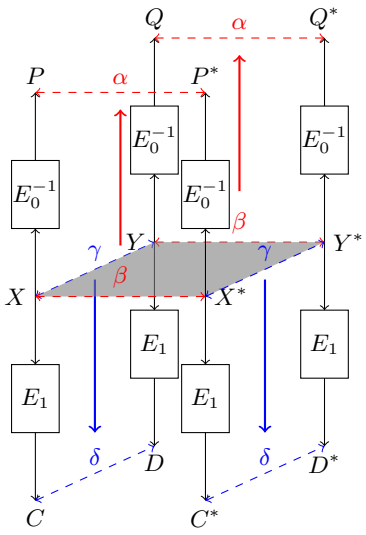

Fig. 1. Schematic view of the attack.

The crucial point is now that such a solution constitutes a second-order differential collision for the block cipher E. We can restate (13) as

$$Q^* - Q - P^* + P = 0 \tag{14}$$

$$E(Q^*) - E(P^*) - E(Q) + E(P) = 0 \tag{15}$$

If we set $\alpha := a_1$ and the difference $Q - P := a_2$ we can rewrite (15) as

$$E(P + a_1 + a_2) - E(P + a_1) - E(P + a_2) + E(P) = 0, \tag{16}$$

that is, we have found a second-order differential collision for the block cipher E. Because of Proposition 1 the same statement is true for the compression function.

2.3 Related Work

The attack presented in this paper stands in relation to previous results in the field of block cipher and hash function cryptanalysis. Figure 1 suggests that it stands between the *boomerang attack* and the *inside-out* attack which were both introduced by Wagner in [28] and also the *rectangle attack* by Biham *et al.* [4]. For the related-key setting, we refer to [5] (among others).

A previous application of the boomerang attack to block-cipher-based hash functions is due to Joux and Peyrin [12], who used the boomerang attack as a neutral bits tool. Another similar attack strategy for hash functions is the *rebound attack* introduced in [20], which was successfully applied to several SHA-3 candidates [17,21] and the ISO/IEC standard Whirlpool [15]. Furthermore, the

second-order differential related-key collisions for the block cipher used in Section 2.2 are called *differential q-multi-collisions* introduced by Biryukov *et al.* in [6] with $q = 2$.

3 Application to SHA-256

In this section, we will show how to construct a second-order differential collision for SHA-256 reduced to 46 (out of 64) steps following the attack strategy described in the previous section. Since the complexity of the attack is quite low, only 2^{46} compression function evaluations, we implemented the attack. An example for a second-order differential collision for SHA-256 reduced to 46 steps is shown in Table 1.

3.1 Previous Results on SHA-256

In the light of the break-through results of Wang *et al.* on the hash functions MD5 and SHA-1, the analysis of SHA-256 is of great interest. In the last few years several cryptanalytic results have been published for SHA-256. In this section, we briefly discuss existing work.

The security of SHA-256 against preimage attacks was first studied by Isobe and Shibutani in [11]. They presented a preimage attack on 24 steps. Later this was improved by Aoki *et al.* to 43 steps in [1]. Both attacks are only slightly faster than the generic attack, which has a complexity of about 2^{256}. In [18], Mendel *et al.* studied the security of SHA-256 with respect to collision attacks. They presented the collision attack on SHA-256 reduced to 19 steps. After that these results have been improved by several researchers. In particular, Nikolic and Biryukov improved in [25] the collision techniques, leading to a collision attack for 23 steps of SHA-256. The best collision attacks so far are extensions of [25]. Indesteege *et al.* [10] and Sanadhya and Sarkar[26], both presented collision attacks for 24 steps. We want to note that in contrast to the preimage attacks all these attacks are of practical complexity. Furthermore, Indesteege *et al.* showed non-random properties for SHA-2 for up to 31 steps. At the rump session of Eurocrypt 2008, Yu and Wang announced that they had shown non-randomness for SHA-256 reduced to 39 steps [31]. In the same presentation they also provided a practical example for 33 steps. However, no details have been published to date. We are not aware of any attack on SHA-256 with practical complexity for more than 33 steps.

3.2 Description of SHA-256

SHA-256 is an iterated hash function that processes 512-bit input message blocks and produces a 256-bit hash value. In the following, we briefly describe the hash function. It basically consists of two parts: the message expansion and the state update transformation. A detailed description of the hash function is given in [23].

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:

$$W_i = \begin{cases} M_i & 0 \leq i < 16 \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} & 16 \leq i < 64 \end{cases} . \quad (17)$$

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

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

State Update Transformation. The state update transformation starts from a (fixed) initial value IV of eight 32-bit words and updates them in 64 steps. In each step one 32-bit word W_i is used to update the state variables $A_i, B_i, \ldots, H_i$. One step of SHA-256 is given in Figure 2.

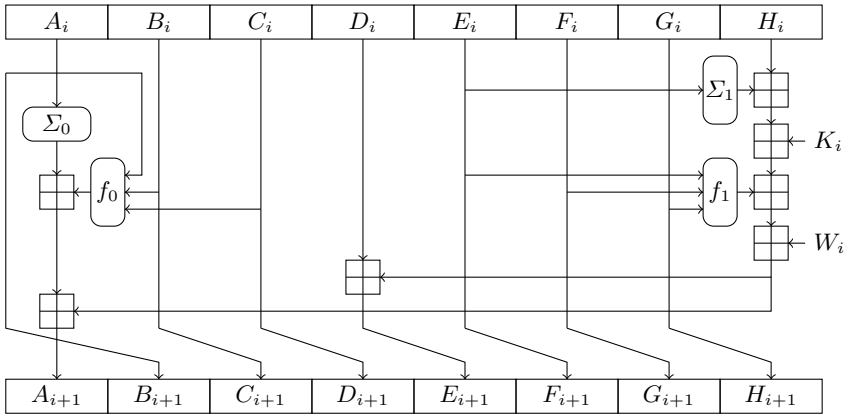

Fig. 2. The step function of SHA-256.

For the definition of the step constants K_i we refer to [23]. The bitwise Boolean functions f_1 and f_0 used in each step are defined as follows:

$$\begin{aligned} f_0(X, Y, Z) &= X \wedge Y \oplus Y \wedge Z \oplus X \wedge Z \\ f_1(X, Y, Z) &= X \wedge Y \oplus \neg X \wedge Z \end{aligned} \quad (19)$$

The linear functions Σ_0 and Σ_1 are defined as follows:

$$\begin{aligned} \Sigma_0(X) &= (X \ggg 2) \oplus (X \ggg 13) \oplus (X \ggg 22) \\ \Sigma_1(X) &= (X \ggg 6) \oplus (X \ggg 11) \oplus (X \ggg 25) \end{aligned} \quad (20)$$

After the last step of the state update transformation, the initial values and the output values of the last step are combined in the feed-forward (Davies-Meyer construction). The result is the final hash value or the initial value for the next message block.

3.3 Differential Characteristics

In this section, we present the differential characteristics used to construct a second-order differential collision for SHA-256 reduced to 46 out of 64 steps. Finding the differential characteristics for both backward and forward direction is the most important and difficult part of the attack. Not only the differential characteristics need to be independent, but also they need to have high probability in order to result in a low attack complexity. As noted before, in general, the assumption on independent characteristics is quite strong, *cf.* [22].

A common approach to construct differential characteristics, which have a high probability, is to use a linearized approximation of the attacked hash function. Finding a differential characteristic in the linearized approximation is not difficult, since it depends only on the differences and not on the actual values. In the case of SHA-256 we approximate all modular additions by the xor operation and the Boolean functions f_0 and f_1 by the 0-function. Using this approximation we found suitable differential characteristics for SHA-256. Note that similar characteristics were used to construct a related-key rectangle distinguisher for 34 steps of the SHACAL-2 block cipher [16].

Table 2. Differential characteristic for steps 1-21 using signed-bit-differences.

backward ↑

i	chaining value	message	prob
0	B: +3 E: +10 -24 -29 H: -12 -17 -23		2^{-10}
1	C: +3 F: +10 -24 -29		2^{-4}
2	D: +3 G: +10 -24 -29		2^{-4}
3	E: +3 H: +10 -24 -29		2^{-7}
4	F: +3		2^{-1}
5	G: +3		2^{-1}
6	H: +3	-3	2^{-1}
7			
⋮	⋮	⋮	⋮
21			

In Table 2 and Table 3 the differential characteristic for both forward and backward direction are shown. Furthermore, the probabilities for each step of the differential characteristics are given. Note that we always assume that the differential characteristic in the message expansion will hold with probability 1. To describe the differential characteristic we use signed-bit differences introduced

by Wang *et al.* in the cryptanalysis of MD5 [30]. The advantage of using signed-bit differences is that there exists a unique mapping to both xor and modular differences. This turned out to be very useful in the attack.

Table 3. Differential characteristic for steps 21-46 using signed-bit-differences. Note that conditions imposed by the characteristic in steps 21-29 are fulfilled in a deterministic way using message modification techniques.

forward

i	chaining value	message	prob
21	B: +6 +9 +18 +20 +25 +29 C: +31 E: -9 -13 -19 F: -18 -29 G: -31 H: +2 -3 +7 +8 +13 -16 -20 +26 +30	-31	2^{-23}
22	C: +6 +9 +18 +20 +25 +29 D: +31 F: -9 -13 -19 G: -18 -29 H: -31		2^{-12}
23	A: -31 D: +6 +9 +18 +20 +25 +29 G: -9 -13 -19 H: -18 -29		2^{-10}
24	B: -31 E: +6 +20 +25 H: -9 -13 -19		2^{-7}
25	C: -31 F: +6 +20 +25		2^{-4}
26	D: -31 G: +6 +20 +25		2^{-4}
27	E: -31 H: +6 +20 +25		2^{-4}
28	F: -31		2^{-1}
29	G: -31		2^{-1}
30	H: -31	+31	1
31			1
⋮	⋮	⋮	⋮
44			1
45		-13 +24 +28	2^{-6}
46	A: -13 +24 +28 E: -13 +24 +28		

message modification

3.4 Complexity of the Attack

Using the differential characteristics given in the previous section, we can construct a second-order differential collision for SHA-256 reduced to 46 out of 64 steps. The differential characteristic used in backward direction holds with probability 2^{-28} and the differential characteristic used in forward direction holds with probability 2^{-72}. Hence, assuming that the two differential characteristics are independent and using the most naive method, *i.e.* random trials, to fulfill all the conditions imposed by the differential characteristics would result in an attack complexity of $2^{2\cdot(72+28)} = 2^{200}$. This is too high for an attack on reduced SHA-256, since the complexity of the generic attack is 2^{85} (see (5)). However, the complexity of the attack can be significantly improved by using message modification techniques. Furthermore, some conditions at the end of the differential characteristics can be ignored which also improves the attack complexity.

Ignoring conditions at the end. As was already observed by Wang *et al.* in the cryptanalysis of SHA-1, conditions resulting from the modular addition in the last steps of the differential characteristic can be ignored [29]. Hence, in the case of the attack on SHA-256, we can ignore 6 conditions in step 46 in the characteristic used in forward direction and 3 conditions in step 1 in the characteristic used in backward direction. This improves the complexity of the attack by a factor of $2^{2\cdot(3+6)} = 2^{18}$ resulting in a complexity of 2^{182}.

Impact of additional less probable characteristics. Even if all message conditions for the two characteristic are already in place, there exist a number of differential characteristics which hold with the same or a slightly lower probability. If also these differential characteristics are allowed, the complexity of the attack can be improved. This has been systematically studied for SHA-1 in [19]. We achieve a significant speedup in the attack on SHA-256 by allowing these additional characteristics. For instance by changing the signs of the differences in chaining variable H_0, we get 2^3 additional differential characteristics for the backward direction which all hold with the same probability as the original differential characteristic given in Table 2. Similarly, we also get 2^3 additional differential characteristic by changing the signs of the differences in chaining variable H_3. This already improves the complexity of the attack by a factor of 2^6. Furthermore, if we do not block the input differences of f_1 and f_0 in step 1, we get 2^4 additional characteristics which again holds with the same probability. Thus, by allowing additional differential characteristics the complexity of the attack can be improved by a factor of 2^{10}, resulting in an attack complexity of 2^{172}. We want to stress, that in practice there exists many more additional differential characteristics that can be used and hence the attack complexity is much lower in practice.

Message modification. As already indicated in Section 2 message modification techniques can be used to significantly improve the complexity of the attack.

The notion of message modification has been introduced by Wang *et al.* in the cryptanalysis of MD5 and other hash functions [30]. The main idea is to choose the message words and internal chaining variables in an attack on the hash function to fulfill the conditions imposed by the differential characteristic in a deterministic way. Luckily, by using message modification techniques, we can fulfill all conditions imposed by the differential characteristic in steps 21-29 by choosing the expanded message words $W_{21}, \ldots, W_{29}$ accordingly. This improves the complexity of the attack by a factor of $2^{2 \cdot 66} = 2^{132}$ resulting in an attack complexity of 2^{40}.

Additional costs coming from the message expansion. So far we assumed that the differential characteristic in the message expansion of SHA-256 will hold with probability 1. However, since the message expansion of SHA-256 is not linear, this is not the case in practice. Indeed most of the conditions that have to be fulfilled to guaranty that the characteristic holds in the message expansion can be fulfilled by choosing the expanded message words and differences in steps 21-29 accordingly. Only the conditions for step 5 and step 6 imposed by the differential characteristic used in backward direction cannot be fulfilled deterministically (see Table 2). In step 6 we need that:

$$W_6^* - W_6 = \texttt{-3} \tag{21}$$

Furthermore, to ensure that there will be no difference in W_5 we need that:

$$W_{21}^* - \sigma_0(W_6^*) - (W_{21} - \sigma_0(W_6)) = \texttt{0} \tag{22}$$

Since (21) will hold with a probability of 2^{-1} and (22) will hold with probability 2^{-2}, this adds an additional factor of $2^{2 \cdot 3} = 2^6$ to the attack complexity. Hence, the final complexity of the attack is 2^{46}.

Implementation. Even though the complexity of the attack was estimated to be about 2^{46}, we expected that the complexity will be lower in practice due to the impact of additional differential characteristics. This was also confirmed by our implementation. In Table 1, an example of a second-order differential collision for 46 steps of SHA-256 is shown.

4 Future Work

4.1 Extending the Attack to More Steps

The attack on 46 steps of SHA-256 can be improved in several ways. First, since the generic complexity to construct a second-order differential collision is 2^{85}, one could extend the attack to more steps by adding steps at the beginning. In other words, the differential characteristic given in Table 2 can be extended by 1 (maybe 2) steps. The result is a theoretical attack for 47 (maybe 48) steps.

Another possibility is to add steps at the end. However, since the backward diffusion is worse than forward diffusion in SHA-256, this does not lead to better results. Second, probably the more interesting approach is to extend the attack to more steps by adding steps in the middle. This can be done by extending the differential characteristic given in Table 3. The advantage of adding steps in the middle instead of adding steps at the beginning or end is that in the current attack only 9 expanded message words are used for message modification (steps 21-29) and hence several expanded message words are left which can still be used for message modification. In other words, by using these expanded message words for message modification, one could extend the attack to more steps by adding steps in the middle without increasing the complexity of the attack. Unfortunately, by adding steps in the middle we could not find differential characteristics for both backward and forward direction anymore which are still independent. However, by using more sophisticated search techniques, like the one invented by De Cannière and Rechberger for SHA-1 [8], one might find such characteristics again. This is part of future work.

4.2 Application to SHA-512

The structure of SHA-512 is very similar to SHA-256. Only the sizes of all words are increased from 32 to 64 bits and the linear functions $\Sigma_0, \Sigma_1, \sigma_0, \sigma_1$ are redefined. Also the number of steps is increased from 64 to 80. Since the design of SHA-512 is very similar to SHA-256 the attack presented in this section for reduced SHA-256 extends in a straight forward way to SHA-512. Furthermore, due to the larger hash size of SHA-512 compared to SHA-256 also the complexity of the generic attack increases, *i.e.* 2^{170} (because of (5)). Hence, the attack can be extended to more steps than it was the case for SHA-256 by adding steps at the beginning. Also, due to the larger word size and hence worse diffusion within the words adding steps in the middle becomes easier. Thus, we expect that several steps can be added in the middle as well. This is work in progress.

5 Conclusion

In this work, we have shown an application of higher-order differential cryptanalysis on block-cipher-based hash functions. Therefore, we adapted several techniques known from block cipher cryptanalysis to hash functions. Applying these techniques to SHA-256 led to an attack for 46 (out of 64) steps of the compression function with practical complexity. The best known attack so far with practical complexity was for 33 steps. Since the structure of SHA-512 and SHA-256 is very similar, the attack transfers to SHA-512 in a straight forward way. Furthermore, due to the larger word size and output size, attacks for more than 46 steps may be expected. In our opinion, the attack seems applicable to a wider range of hash function constructions. In particular, several of the SHA-3 candidates seem to be a natural target for this attack.

Acknowledgements

The work in this paper has been supported in part by the Secure Information Technology Center - Austria (A-SIT), by the Austrian Science Fund (FWF), project P21936-N23 and by the European Commission under contract ICT-2007-216646 (ECRYPT II).

References

1. Kazumaro Aoki, Jian Guo, Krystian Matusiewicz, Yu Sasaki, and Lei Wang. Preimages for Step-Reduced SHA-2. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *LNCS*, pages 578–597. Springer, 2009.
2. Jean-Philippe Aumasson, Itai Dinur, Willi Meier, and Adi Shamir. Cube Testers and Key Recovery Attacks on Reduced-Round MD6 and Trivium. In Orr Dunkelman, editor, *FSE*, volume 5665 of *LNCS*, pages 1–22. Springer, 2009.
3. Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. On the Indifferentiability of the Sponge Construction. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *LNCS*, pages 181–197. Springer, 2008.
4. Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack - Rectangling the Serpent. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *LNCS*, pages 340–357. Springer, 2001.
5. Eli Biham, Orr Dunkelman, and Nathan Keller. Related-Key Boomerang and Rectangle Attacks. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 507–525. Springer, 2005.
6. Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and Related-Key Attack on the Full AES-256. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *LNCS*, pages 231–249. Springer, 2009.
7. Christina Boura and Anne Canteaut. Zero-Sum Distinguishers for Iterated Permutations and Applications to Keccak-f and Hamsi-256. In *Selected Areas in Cryptography*, LNCS. Springer, 2010. To appear.
8. Christophe De Cannière and Christian Rechberger. Finding SHA-1 Characteristics: General Results and Applications. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT*, volume 4284 of *LNCS*, pages 1–20. Springer, 2006.
9. Itai Dinur and Adi Shamir. Cube Attacks on Tweakable Black Box Polynomials. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *LNCS*, pages 278–299. Springer, 2009.
10. Sebastiaan Indesteege, Florian Mendel, Bart Preneel, and Christian Rechberger. Collisions and Other Non-random Properties for Step-Reduced SHA-256. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography*, volume 5381 of *LNCS*, pages 276–293. Springer, 2008.
11. Takanori Isobe and Kyoji Shibutani. Preimage Attacks on Reduced Tiger and SHA-2. In Orr Dunkelman, editor, *FSE*, volume 5665 of *LNCS*, pages 139–155. Springer, 2009.
12. Antoine Joux and Thomas Peyrin. Hash Functions and the (Amplified) Boomerang Attack. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *LNCS*, pages 244–263. Springer, 2007.
13. Lars R. Knudsen. Truncated and Higher Order Differentials. In Bart Preneel, editor, *FSE*, volume 1008 of *LNCS*, pages 196–211. Springer, 1994.

14. Xuejia Lai. Higher Order Derivatives and Differential Cryptanalysis. In Richard E. Blahut, Daniel J. Costello Jr., Ueli Maurer, and Thomas Mittelholzer, editors, *Communications and Cryptography: Two Sides of One Tapestry*, pages 227–233. Kluwer Academic Publishers, 1994.
15. Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *LNCS*, pages 126–143. Springer, 2009.
16. Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Related-Key Rectangle Attack on 42-Round SHACAL-2. In Sokratis K. Katsikas, Javier Lopez, Michael Backes, Stefanos Gritzalis, and Bart Preneel, editors, *ISC*, volume 4176 of *LNCS*, pages 85–100. Springer, 2006.
17. Krystian Matusiewicz, María Naya-Plasencia, Ivica Nikolic, Yu Sasaki, and Martin Schläffer. Rebound Attack on the Full Lane Compression Function. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *LNCS*, pages 106–125. Springer, 2009.
18. Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Analysis of Step-Reduced SHA-256. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *LNCS*, pages 126–143. Springer, 2006.
19. Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. The Impact of Carries on the Complexity of Collision Attacks on SHA-1. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *LNCS*, pages 278–292. Springer, 2006.
20. Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In Orr Dunkelman, editor, *FSE*, volume 5665 of *LNCS*, pages 260–276. Springer, 2009.
21. Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. Rebound Attacks on the Reduced Grøstl Hash Function. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *LNCS*, pages 350–365. Springer, 2010.
22. Sean Murphy. The Return of the Boomerang. *IEEE Transactions on Information Theory*, 2010. To appear. `http://www.isg.rhul.ac.uk/~sean/CLN9-909_IEEE_Final0.pdf`.
23. National Institute of Standards and Technology. FIPS PUB 180-2: Secure Hash Standard. Federal Information Processing Standards Publication 180-2, U.S. Department of Commerce, August 2002. Available online: `http://www.itl.nist.gov/fipspubs`.
24. National Institute of Standards and Technology (NIST). Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family. *Federal Register*, 27(212):62212–62220, November 2007. Available online: `http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf`.
25. Ivica Nikolic and Alex Biryukov. Collisions for Step-Reduced SHA-256. In Kaisa Nyberg, editor, *FSE*, volume 5086 of *LNCS*, pages 1–15. Springer, 2008.
26. Somitra Kumar Sanadhya and Palash Sarkar. New Collision Attacks against Up to 24-Step SHA-2. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT*, volume 5365 of *LNCS*, pages 91–103. Springer, 2008.
27. Michael Vielhaber. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive, Report 2007/413, 2007. `http://eprint.iacr.org/`.
28. David Wagner. The Boomerang Attack. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *LNCS*, pages 156–170. Springer, 1999.

29. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *LNCS*, pages 17–36. Springer, 2005.
30. Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005.
31. Xiaoyun Wang and Hongbo Yu. Non-randomness of 39-step SHA-256. Presented at rump session of EUROCRYPT, 2008.
32. Dai Watanabe, Yasuo Hatano, Tsuyoshi Yamada, and Toshinobu Kaneko. Higher Order Differential Attack on Step-Reduced Variants of *Luffa* v1. In Seokhie Hong and Tetsu Iwata, editors, *FSE*, volume 6147 of *LNCS*, pages 270–285. Springer, 2010.