Second-Order Differential Collisions for Reduced SHA-256

Alex Biryukov[1], Mario Lamberger[2], Florian Mendel[2], and Ivica Nikolić[1]

[1] University of Luxembourg, Luxembourg
[2] IAIK, Graz University of Technology, Austria

Abstract. In this work, we introduce a new non-random property for hash/compression functions using the theory of higher order differentials. Based on this, we show a second-order differential collision for the compression function of SHA-256 reduced to 47 out of 64 steps with practical complexity. We have implemented the attack and provide an example. Our results suggest that the security margin of SHA-256 is much lower than the security margin of most of the SHA-3 finalists in this setting. The techniques employed in this attack are based on a rectangle/boomerang approach and cover advanced search algorithms for good characteristics and message modification techniques. Our analysis also exposes flaws in all of the previously published related-key rectangle attacks on the SHACAL-2 block cipher, which is based on SHA-256. We provide valid rectangles for 48 steps of SHACAL-2.

Keywords: Hash functions, higher-order differentials, non-randomness, SHA-256, SHACAL-2

1 Introduction

The significant advances in the field of hash function research that have been made in the recent years, had a formative influence on the landscape of hash functions. The analysis of MD5 and SHA-1 has convinced many cryptographers that these widely deployed hash functions can no longer be considered secure [39,40]. As a consequence, people are evaluating alternative hash functions in the SHA-3 initiative organized by NIST [29]. During this ongoing evaluation, not only the three classical security requirements (preimage resistance, 2nd preimage resistance and collision resistance) are considered. Researchers look at (semi-) free-start collisions, near-collisions, etc. Whenever a behavior different from the one expected of a 'random oracle' can be demonstrated for a new hash function, it is considered suspect, and so are the weaknesses that are demonstrated only for the compression function. In light of this, for four out of the five third round SHA-3 candidates the best attacks are in the framework of distinguishers: boomerang distinguisher for BLAKE [6], differential distinguisher for Grøstl [32], zero-sum distinguisher on Keccak [8] and rotational rebound distinguisher for Skein [17].

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 [13,33] and a preimage attack on 45 steps [18] having a complexity of $2^{255.5}$.

Higher-order differentials have been introduced by Lai in [21] and first applied to block ciphers by Knudsen in [20]. The application to stream ciphers was proposed by Dinur and Shamir in [10] and Vielhaber in [35]. First attempts to apply these strategies to hash functions were published in [2]. Recently, higher-order differential attacks have been applied to several hash functions submitted to the SHA-3 initiative organized by NIST such as BLAKE [6], Hamsi [7], Keccak [8], and Luffa [42].

In this work, we present a second-order differential collision for the SHA-256 compression function on 47 out of 64 steps having practical complexity. The attack is an application of higher-order differentials on hash functions. Table 3 shows the resulting example.

Since our attack technique resembles boomerang/rectangle attacks, known from the cryptanalysis of block ciphers, we use a strict criterion for checking that the switch in the middle does not contain any contradictions that can appear due to the independency assumption of the characteristics used in the rectangle. We show that all the previous related-key rectangle distinguishers for SHACAL-2 have a common flaw in the switch due to these assumptions and present a rectangle distinguisher for 48 steps that passes our check.

Our analysis shows that the compression functions exhibit non-random properties, though they do not lead to collision/preimage attacks on the hash functions. Nevertheless, the attacks give a clear indication that if we compare the security of SHA-256 to the security of the third round SHA-3 candidates, in the this setting, then SHA-256 has one of the lowest security margins.

2 Higher-Order Differential Collisions for Compression 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 [21] and subsequently applied by Knudsen in [20]. We recall the basic definitions that we will use in the subsequent sections.

Definition 1. *Let $(S,+)$ and $(T,+)$ be abelian groups. For a function $f\colon S \to T$, 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 a function $f\colon S \to T$ 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 $f\colon S \to T$ 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 $i = 1$.

In this work, we concentrate on *second-order differential collisions*, *i.e.* $i = 2$:

$$f(y) - f(y + a_2) + f(y + a_1 + a_2) - f(y + a_1) = 0 \tag{5}$$

Further we assume that we have oracle access to a function $f\colon S \to T$ and measure the complexity in the number of queries to f, *i.e. query complexity*, while ignoring all other computations, memory accesses, etc. Additionally, we will restrict ourselves to functions f mapping to groups $(T, +)$ with $|T| = 2^n$ which are endowed with an additive operation.

Definition 4. *Let $f\colon S \to T$ be as above. A solution $(y, a_1, a_2) \in S^3$ to (5) is called trivial if the complexity of producing it is $O(1)$, otherwise it is called non-trivial.*

Lemma 1. *Let $f\colon S \to T$ be as above. Then, a trivial solution to (5) can be found if*

1. *f is linear, or*
2. *at least one of a_1, a_2 is zero, or*
3. *$a_1 = a_2$ and the group $(T, +)$ is of the form*

$$(T, +) \simeq (\mathbb{Z}_2, +)^{n-\ell} \oplus (\mathbb{Z}_{2^\ell}, +), \tag{6}$$

for small ℓ.

Proof. If f is a linear function, then (5) collapses and any choice of (y, a_1, a_2) is a valid solution. Under the assumption that f is drawn uniformly at random from all functions $f\colon S \to T$, and T is not as in (6), then the only trivial solution to equation (5) is when the inputs coincide, *i.e.* either $y = y + a_2$ and

$y + a_1 + a_2 = y + a_1$ leading to the case where $a_2 = 0$, or $y = y + a_1$ and $y + a_1 + a_2 = y + a_2$ leading to $a_1 = 0$.

In the third case, equation (5) boils down to $2f(y) = 2f(y + a)$. In general this is a classical meet-in-the-middle problem, however if $(T, +)$ is as in (6), this equation holds with a probability $2^{-2(\ell-1)}$ for a random function f which leads to trivial solutions for small values of ℓ.

For all the other cases, the problem of finding a solution is an instance of the generalized birthday problem proposed by Wagner [37] and therefore the number of queries depends on n. ∎

We now want to lower bound the query complexity of producing a non-trivial differential collision of order 2.

Theorem 1. *For a function $f\colon S \to T$ with $|T| = 2^n$, the query complexity for producing a non-trivial differential collision of order 2 is $\Omega(2^{n/3})$.*

Proof. To find an input (y, a_1, a_2) such that (5) holds, one has to try around 2^n different tuples – otherwise the required value 0, may not appear. We can freely choose three input parameters, *i.e.* y, a_1, a_2, which then fix the remaining one. Therefore, (5) can be split into three parts (but not more!), and solved by generating three independent lists of values. Obviously, the number of queries is the lowest when these lists have equal size. Hence, to have a solution for (5), one has to choose $2^{n/3}$ values for each of y, a_1, a_2, and therefore the query complexity of a differential collision of order 2 for f is $\Omega(2^{n/3})$. ∎

Remark 1. We want to note that the actual complexity might be much higher in practice than this bound for the query complexity. We are not aware of any algorithm faster than $2^{n/2}$, since dividing (5) into three independent parts is not possible (one of the terms has all the inputs, and any substitution of variables leads to a similar case).

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

In all of the following, we consider block ciphers $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ 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,

$$(\{0,1\}^n, +) := \underbrace{(\mathbb{Z}_{2^w}, +) \times \cdots \times (\mathbb{Z}_{2^w}, +)}_{\ell \text{ times}}. \tag{7}$$

The latter definition clearly aims very specifically at the SHA-2 design. 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:

$$x_i = E(m_i, x_{i-1}) + x_{i-1} \tag{8}$$

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 (8) 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:

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

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*

$$f(y) = E(y) + L(y), \tag{10}$$

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

$$\deg(\Delta_{(a)} f(y)) \leq \deg(f(y)) - 1\,. \tag{11}$$

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 (10) 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. ∎

Hence, if we want to construct a second order collision for the compression function f it is sufficient to construct a second-order collision for the block cipher. 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 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 2.2 for related work. We also want to stress, that due to our definition above, the following differentials are actually related-key differentials. We have

$$E_0^{-1}(y + \beta) - E_0^{-1}(y) = \alpha \tag{12}$$

and

$$E_1(y + \gamma) - E_1(y) = \delta \tag{13}$$

where the differential in E_0^{-1} holds with probability p_0 and in E_1 holds with probability p_1. Using these two 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 (12) and (13),

$$P^* - P = Q^* - Q = \alpha, \text{ resp. } D - C = D^* - C^* = \delta, \tag{14}$$

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.* [28]. However, if this assumption holds, the expected number of solutions to (14) 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.* [40]) can be used to improve this complexity.

The crucial point now is that such a solution constitutes a second-order differential collision for the block cipher E. We can restate (14) as

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

$$E(Q^*) - E(P^*) - E(Q) + E(P) = 0 \tag{16}$$

If we set $\alpha := a_1$ and the difference $Q - P := a_2$ we can rewrite (16) as

$$E(P + a_1 + a_2) - E(P + a_1) - E(P + a_2) + E(P) = 0, \tag{17}$$

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.

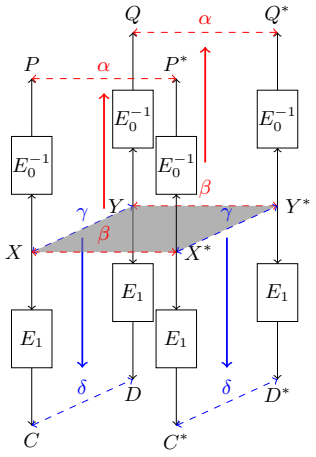

Fig. 1. Schematic view of the attack.

2.2 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 [36] and also the *rectangle attack* by Biham *et al.* [3]. For the related-key setting, we refer to [4] (among others). We also want to refer to the *amplified boomerang attack* [16]. A previous application of the boomerang attack to block-cipher-based hash functions is due to Joux and Peyrin [15], who used the boomerang attack as a neutral bits tool. Another similar attack strategy for hash functions is the *rebound attack* introduced in [27]. Furthermore, the second-order differential related-key collisions for the block cipher used in Section 2.1 are called *differential q-multi-collisions* introduced by Biryukov *et al.* in [5] with $q = 2$. Recently, an attack framework similar to this was proposed in [6,22] and applied to HAVAL in [34].

3 Application to SHA-256

In the light of the breakthrough results of Wang *et al.* on the hash functions MD5 and SHA-1, the analysis of SHA-256 is of great interest. Moreover, SHA-2 is a reference point in terms of speed but also security for the SHA-3 candidates.

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 [14]. They presented a preimage attack on 24 steps. This was improved by Aoki *et al.* to 43 steps in [1] and later extended to 45 steps by Khovratovich *et al.* in [18]. All attacks are only slightly faster than the generic attack, which has a complexity of about 2^{256}. In [25], Mendel *et al.* studied the security of SHA-256 with respect to collision attacks. They presented the collision attack on SHA-256 reduced to 18 steps. After that these results have been improved by several researchers. In particular, Nikolić and Biryukov improved in [31] the collision techniques, leading to a collision attack for 23 steps of SHA-256. The best collision attacks so far are extensions of [31]. Indesteege *et al.* [13] and Sanadhya and Sarkar[33], 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 [41]. 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. In this section, we show how to construct a second-order differential collision for SHA-256 reduced to 47 (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 of a second-order differential collision for SHA-256 reduced to 47 steps is shown in Table 3.

3.1 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 [30].

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:

$$W_i = \begin{cases} M_i & 0 \leq i < 16 \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} & 16 \leq i < 64 \end{cases} . \qquad (18)$$

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

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

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 $A_i, B_i, \ldots, H_i$ as follows:

$$\begin{aligned} T_1 &= H_i + \Sigma_1(E_i) + f_1(E_i, F_i, G_i) + K_i + W_i \ , \\ T_2 &= \Sigma_0(A_i) + f_0(A_i, B_i, C_i) \ , \\ A_{i+1} &= T_1 + T_2 \ , \ B_{i+1} = A_i \ , \ C_{i+1} = B_i \ , \ D_{i+1} = C_i \ , \\ E_{i+1} &= D_i + T_1 \ , \ F_{i+1} = E_i \ , \ G_{i+1} = F_i \ , \ H_{i+1} = G_i \ . \end{aligned} \tag{20}$$

For the definition of the step constants K_i we refer to [30]. The bitwise Boolean functions f_1 and f_0 used in each step are defined as follows:

$$\begin{aligned} f_0(X, Y, Z) &= X \wedge Y \oplus Y \wedge Z \oplus X \wedge Z \\ f_1(X, Y, Z) &= X \wedge Y \oplus \neg X \wedge Z \end{aligned} \tag{21}$$

The linear functions Σ_0 and Σ_1 are defined as follows:

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

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

3.2 Differential Characteristics

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.* [28].

We apply a particular approach to construct differential characteristics that are used to construct second-order differential collisions for reduced SHA-256. We run a full search for *sub-optimal* differential characteristics, *i.e.* characteristics with the following properties:

- use a linearized approximation of the attacked hash function, *i.e.* approximate all modular additions by the xor operation;
- approximate the Boolean functions f_0 and f_1 by the 0-function, except in the bits j, where either $\Delta A[j] = \Delta B[j] = \Delta C_i[j] = 1$ or $\Delta F[j] = \Delta G[j] = 1$ – in these bits approximate with 1. This requirement comes from the fact that

if all three inputs to f_0 have a difference, then the output has a difference (with probability 1); a similar property holds for f_1. Note that it is possible to approximate some bits with either 0 or 1, however, this introduces a high branching leading to an infeasible search;
– the characteristic has a single bit difference in the message word at some step i ($i \leq 16$), followed by 15 message words without difference. When using such characteristic, 16 steps (the ones that follow i) can be passed with probability 1 – arguably, any characteristic that does not follow this strategy will have a low probability due to the fast diffusion of the difference coming from the message words. This type of characteristics was used to construct various related-key rectangle distinguishers for SHACAL-2 [11,19,23,24,38].

Once we have the set of sub-optimal characteristics, we try to combine them for the second-order differential collision scenario, *i.e.* try to check if the switch in the middle is possible. This is a very important requirement, as some of the characteristics cannot be combined, *i.e.* their combination leads to contradictions. Some of the conditions for the switch can be checked only by examining the differences in the characteristics, while other are checked by confirming experimentally the absence of contradictions in the switch.

Table 1. Differential characteristic for steps 1-22 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			1
⋮	⋮	⋮	⋮
20			1
21		+17 +28	1
22	A: +17 +28 E: +17 +28		

In Table 1 and Table 2 the differential characteristics for both forward and backward direction are shown. Furthermore, the probabilities for each step of the differential characteristics are given. Note that for start we 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 [40]. The advantage of using signed-bit differences is that there exists a unique mapping to both xor and modular differences. Another advantage is that the feedforward in SHA-256 is modular, hence no additional probability will be introduced for this operation.

Table 2. Differential characteristic for steps 23-47 using signed-bit-differences. Note that conditions imposed by the characteristic in steps 23-30 are fulfilled in a deterministic way using message modification techniques.

i	chaining value	message	prob
22	B: +3 +12 +14 +19 +23 +32 C: +25 E: -3 -7 -13 F: -12 -23 G: -25 H: -1 +3 +7 +14 +15 +24 +26 +28 -30	-25	2^{-22}
23	C: +3 +12 +14 +19 +23 +32 D: +25 F: -3 -7 -13 G: -12 -23 H: -25		2^{-13}
24	A: -25 D: +3 +12 +14 +19 +23 +32 G: -3 -7 -13 H: -12 -23		2^{-10}
25	B: -25 E: +14 +19 +32 H: -3 -7 -13		2^{-7}
26	C: -25 F: +14 +19 +32		2^{-4}
27	D: -25 G: +14 +19 +32		2^{-4}
28	E: -25 H: +14 +19 +32		2^{-4}
29	F: -25		2^{-1}
30	G: -25		2^{-1}
31	H: -25	+25	1
32			1
⋮	⋮	⋮	⋮
45			1
46		-7 -18 -22	2^{-6}
47	A: -7 -18 -22 E: -7 -18 -22		

forward

message modification

3.3 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 47 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 a practical attack on reduced SHA-256. However, the complexity can be significantly reduced by using message modification techniques. Moreover, 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 in the cryptanalysis of SHA-1, conditions resulting from the modular addition in the last steps of the differential characteristic can be ignored [9,39]. The reason is that we do not care

about carries in the last steps, since the modular difference will be the same. In 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 the message conditions for the two characteristics are already in place, there exist a number of differential characteristics which hold with the same or a slightly lower probability. Hence, it is advantageous to consider differentials. A similar effect has been exploited by Kelsey *et al.* in the amplified boomerang attack on block ciphers [16]. For hash functions, this has been systematically studied for SHA-1 in [26]. 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 1. 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 exist 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 [40]. 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 22-30 by choosing the expanded message words $W_{22}, \ldots, W_{30}$ 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 21-30 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 1). In step 6 we need that:

$$W_6^* - W_6 = 3 \tag{23}$$

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

$$W_{21}^* - \sigma_0(W_6^*) - (W_{21} - \sigma_0(W_6)) = 0 \tag{24}$$

Since (23) will hold with a probability of 2^{-1} and (24) 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}. By Theorem 1, the complexity in the generic case is around 2^{85}.

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 confirmed by our implementation. In Table 3, an example of a second-order differential collision for 47 steps of SHA-256 is shown.

Table 3. 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 47 steps of the SHA-256 compression function.

y	89456784 4ef9daf6 0ab509f5 3fdf6c93 fe7afc67 b03ad81a fd306df9 1d14cadd
	daea3041 70f45fd7 4a03bf20 c13c961c 6a12c686 fc7be50c 7b060fc2 0ee1e276
	630c3c7e 734246a4 88401eb0 9aac88c1 4b6bca45 b777c1e6 5537cdb1 9b5bc93b
a_1	00000000 00000000 00000000 00000000 00000000 00000000 00000004 00000000
	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
	00000000 fffffffc 00000000 fffffffc 10800200 00000000 ff800000 803ef414
a_2	2335e851 20f48326 69151911 f5cb76c2 b9d69e31 32685b9c 90cceff7 081ebbf7
	967c8864 a43138d1 7e9a3eec c39cf7d3 5914e008 8d0d3b73 e077c63f d29db1b0
	742b8c01 92248811 a119f182 dd829be5 e3e1802e 21130e9f 1dacd7d3 8acf11fe

4 Applications to Related Primitives

The results presented in the previous section have a direct impact on the analysis of primitives similar to SHA-256. First of all, due to the similar design of SHA-256 and SHA-512 the attack extends in straight forward way to SHA-512. Second, our search for sub-optimal characteristics in SHA-256, can be used to find suitable characteristics for a related-key rectangle attack on the SHACAL-2 block cipher [12]. which is based on SHA-256. The block cipher proposed by Handschuh and Naccache in 2000 and was evaluated by NESSIE.

4.1 Application to SHA-512

The structure of SHA-512 is very similar to SHA-256 – only the size of the words is 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 for 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.* it becomes around 2^{170}. 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.

4.2 Application to SHACAL-2

In the past several related-key rectangle attacks have been published for the SHACAL-2 block cipher [11,19,23,24,38]. It is interesting to note that all of the published rectangles on SHACAL-2 contain a flaw in the analysis. This flaw is in the switch of the rectangle, since the used characteristics are not independent and the conditions cannot be satisfied simultaneously in both of the characteristics. In the rectangles in [24,38,11], in the the switch in the middle the following differences in bit 13 are defined: at the output of the backward characteristic $\Delta E[13] = 1$, $\Delta F[13] = \Delta G[13] = 0$; at the input of the forward characteristic $\Delta E[13] = 0, \Delta F[13] = 1, \Delta G[13] = 0$. At the first step of the forward characteristics it is assumed that the output difference of f_1 is zero. However, this is not possible for both of the characteristics. Since $\Delta F[13] = 1$, the value of $E[13]$ has to be 0. Then, in the second characteristic (on the other side of the rectangle), since the output difference $\Delta E[13]$ is 1, then this $E[13]$ will be 1, and therefore the output of f_1 in bit 13 will produce difference. A similar contradiction can be seen in [23]. First, since there is a difference in bit 13 in E_{25} coming from the upper trail, one needs the differences in F_{25} and G_{25} in bit 13 to be the same (have the same sign) in the lower trail (see Table 3), otherwise there will be a contradiction. In the next step we have $G_{26} = F_{25}$ and $H_{26} = G_{25}$ and hence the difference in bit 13 of G_{26} and H_{26} have the same sign. This leads now to a contradiction, since in the characteristic it is required that these two differences cancel out. However, since they have the same sign this is not possible and we get a contradiction. In [19], in the lower trail (Table 6) there are conditions on E_{24} in bits (2,14,15,24,25) to guarantee that the differences in G_{24} behave correctly, in particular the bit 24 of E_{24} has to be 1. But from the upper trail we get difference in W_{23} in bits 13,24, and 28, and hence E_{24} will have difference in bits 13,24,28. Therefore, E_{24} cannot take the value 1 (in these three bits) in both of the bottom characteristics. This can be fixed by allowing a carry from bit 13 to 24 to cancel the difference in bit 24, but then there will always be a difference in bit 14 and 15 which again leads to a contradiction.

Table 4. Differential characteristic using xor-differences for the rectangle distinguisher on 48 steps of SHACAL-2.

i	chaining value	message	prob
0	C: 32 28 23 21 12 9 D: 2 F: 22 16 12 G: 32 21 12 H: 12 2		2^{-15}
1	A: 2 D: 32 28 23 21 12 9 G: 22 16 12 H: 32 21 12		2^{-11}
2	B: 2 E: 28 23 9 H: 22 16 12		2^{-7}
3	C: 2 F: 28 23 9		2^{-4}
4	D: 2 G: 28 23 9		2^{-4}
5	E: 2 H: 28 23 9		2^{-4}
6	F: 2		2^{-1}
7	G: 2		2^{-1}
8	H: 2	2	2^{-1}
9			1
⋮	⋮	⋮	⋮
22			1
23		27 16	2^{-4}
24	A: 27 16 E: 27 16		

i	chaining value	message	prob
24	C: 30 26 21 19 10 7 D: 32 F: 20 14 10 G: 30 19 10 H: 32 10		2^{-13}
25	A: 32 D: 30 26 21 19 10 7 G: 20 14 10 H: 30 19 10		2^{-13}
26	B: 32 E: 26 21 7 H: 20 14 10		2^{-7}
27	C: 32 F: 26 21 7		2^{-4}
28	D: 32 G: 26 21 7		2^{-3}
29	E: 32 H: 26 21 7		2^{-4}
30	F: 32		2^{-1}
31	G: 32		2^{-1}
32	H: 32	32	1
33			1
⋮	⋮	⋮	⋮
46			1
47		29 25 14	2^{-6}
48	A: 29 25 14 E: 29 25 14		

Each of the published rectangle attack works for the whole key space. Further, we relax this requirement, *i.e.* we examine the security of the cipher in a weak-key class. These types of attacks are inline with the recent attacks on AES-256 [5]. We analyze a secret-key primitive, hence the message modification techniques presented in the previous section are not applicable and therefore the complexity of the attack is fully determined by the probability of the characteristics used in the rectangle. The probability of the characteristic in the key schedule not necessarily has to be 1 (it is a weak-key class), however this adds to the total complexity of the attack.

Our search for sub-optimal characteristics in SHA-256, can be used as well to find characteristics suitable for a related-key rectangle attack on SHACAL-2. Note that the search avoids using the above mentioned characteristics (with flaws), since it checks experimentally, that all the conditions on the switch can be satisfied.

We found a 48-step related-key rectangle distinguisher with two different characteristics, the first on 24 steps with 2^{-52}, and the second on 24 steps with 2^{-52} (see Table 4). The probability of the key schedule (message expansion) is $2^{-8.5}$. Therefore, the total probability of the rectangle is $2^{-216.5}$. Using some available techniques, e.g. the one presented in [24], we can add one step at the beginning, and two steps at the end of the rectangle, to obtain a key recovery attack on 51 steps of SHACAL-2.

5 Conclusions

In this work, we have shown an application of higher-order differential cryptanalysis on block-cipher-based hash functions. In our attack, we adapted several techniques known from block cipher cryptanalysis to hash functions. Applying these techniques to SHA-256 led to an attack for 47 (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 steps may be expected. We also want to note that the attacks cannot be extended to the hash function to construct collisions or (second) preimages.

However, based on our results, a few conclusions can be deduced. First, SHA-256 has a low security margin against practical distinguishers. Its compression function seems to be weaker than those of the third round SHA-3 candidates, as none of them has practical distinguishers covering such a high percentage of the total number of steps.

Second, when applying boomerang/rectangle attacks to word oriented primitives, the switch in the middle has to be checked carefully – the flaws we have presented as well as our experiments indicate that only a very small percentage of characteristics (even with sparse input-output differences) can be combined.

Finally, the basic strategy described in this paper, *i.e.* linearize the compression function, search for sub-optimal characteristics and combine them in a boomerang/rectangle attack, can be used as a preliminary security analysis for hash functions in general.

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. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for Step-Reduced SHA-2. In: Matsui, M. (ed.) ASIACRYPT. LNCS, vol. 5912, pp. 578–597. Springer (2009)
2. Aumasson, J.P., Dinur, I., Meier, W., Shamir, A.: Cube Testers and Key Recovery Attacks on Reduced-Round MD6 and Trivium. In: Dunkelman, O. (ed.) FSE. LNCS, vol. 5665, pp. 1–22. Springer (2009)
3. Biham, E., Dunkelman, O., Keller, N.: The Rectangle Attack - Rectangling the Serpent. In: Pfitzmann, B. (ed.) EUROCRYPT. LNCS, vol. 2045, pp. 340–357. Springer (2001)
4. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In: Cramer, R. (ed.) EUROCRYPT. LNCS, vol. 3494, pp. 507–525. Springer (2005)

5. Biryukov, A., Khovratovich, D., Nikolić, I.: Distinguisher and Related-Key Attack on the Full AES-256. In: Halevi, S. (ed.) CRYPTO. LNCS, vol. 5677, pp. 231–249. Springer (2009)
6. Biryukov, A., Nikolić, I., Roy, A.: Boomerang Attacks on BLAKE-32. In: Joux, A. (ed.) FSE. LNCS, vol. 6733, pp. 218–237. Springer (2011)
7. Boura, C., Canteaut, A.: Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak- and Hamsi-256. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC. LNCS, vol. 6544, pp. 1–17. Springer (2010)
8. Boura, C., Canteaut, A., De Cannière, C.: Higher-Order Differential Properties of Keccak and Luffa. In: Joux, A. (ed.) FSE. LNCS, vol. 6733, pp. 252–269. Springer (2011)
9. De Cannière, C., Mendel, F., Rechberger, C.: Collisions for 70-Step SHA-1: On the Full Cost of Collision Search. In: Adams, C.M., Miri, A., Wiener, M.J. (eds.) SAC. LNCS, vol. 4876, pp. 56–73. Springer (2007)
10. Dinur, I., Shamir, A.: Cube Attacks on Tweakable Black Box Polynomials. In: Joux, A. (ed.) EUROCRYPT. LNCS, vol. 5479, pp. 278–299. Springer (2009)
11. Fleischmann, E., Gorski, M., Lucks, S.: Memoryless Related-Key Boomerang Attack on 39-Round SHACAL-2. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC. LNCS, vol. 5451, pp. 310–323. Springer (2009)
12. Handschuh, H., Naccache, D.: SHACAL. Submitted as an NESSIE Candidate Algorithm (2000), available online: `http://www.cryptonessie.org`
13. 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. LNCS, vol. 5381, pp. 276–293. Springer (2008)
14. Isobe, T., Shibutani, K.: Preimage Attacks on Reduced Tiger and SHA-2. In: Dunkelman, O. (ed.) FSE. LNCS, vol. 5665, pp. 139–155. Springer (2009)
15. Joux, A., Peyrin, T.: Hash Functions and the (Amplified) Boomerang Attack. In: Menezes, A. (ed.) CRYPTO. LNCS, vol. 4622, pp. 244–263. Springer (2007)
16. Kelsey, J., Kohno, T., Schneier, B.: Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In: Schneier, B. (ed.) FSE. LNCS, vol. 1978, pp. 75–93. Springer (2000)
17. Khovratovich, D., Nikolić, I., Rechberger, C.: Rotational Rebound Attacks on Reduced Skein. In: Abe, M. (ed.) ASIACRYPT. LNCS, vol. 6477, pp. 1–19. Springer (2010)
18. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 family. Cryptology ePrint Archive, Report 2011/286 (2011)
19. Kim, J., Kim, G., Lee, S., Lim, J., Song, J.H.: Related-Key Attacks on Reduced Rounds of SHACAL-2. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT. LNCS, vol. 3348, pp. 175–190. Springer (2004)
20. Knudsen, L.R.: Truncated and Higher Order Differentials. In: Preneel, B. (ed.) FSE. LNCS, vol. 1008, pp. 196–211. Springer (1994)
21. Lai, X.: Higher Order Derivatives and Differential Cryptanalysis. In: Blahut, R.E., Costello Jr., D.J., Maurer, U., Mittelholzer, T. (eds.) Communications and Cryptography: Two Sides of One Tapestry. pp. 227–233. Kluwer Academic Publishers (1994)
22. Lamberger, M., Mendel, F.: Higher-Order Differential Attack on Reduced SHA-256. Cryptology ePrint Archive, Report 2011/037 (2011)
23. Lu, J., Kim, J.: Attacking 44 Rounds of the SHACAL-2 Block Cipher Using Related-Key Rectangle Cryptanalysis. IEICE Transactions 91-A(9), 2588–2596 (2008)

24. Lu, J., Kim, J., Keller, N., Dunkelman, O.: Related-Key Rectangle Attack on 42-Round SHACAL-2. In: Katsikas, S.K., Lopez, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC. LNCS, vol. 4176, pp. 85–100. Springer (2006)
25. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: Analysis of Step-Reduced SHA-256. In: Robshaw, M.J.B. (ed.) FSE. LNCS, vol. 4047, pp. 126–143. Springer (2006)
26. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: The Impact of Carries on the Complexity of Collision Attacks on SHA-1. In: Robshaw, M.J.B. (ed.) FSE. LNCS, vol. 4047, pp. 278–292. Springer (2006)
27. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: Dunkelman, O. (ed.) FSE. LNCS, vol. 5665, pp. 260–276. Springer (2009)
28. Murphy, S.: The Return of the Cryptographic Boomerang. IEEE Transactions on Information Theory 57(4), 2517–2521 (2011)
29. National Institute of Standards and Technology: Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family. Federal Register 27(212), 62212–62220 (November 2007)
30. 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)
31. Nikolić, I., Biryukov, A.: Collisions for Step-Reduced SHA-256. In: Nyberg, K. (ed.) FSE. LNCS, vol. 5086, pp. 1–15. Springer (2008)
32. Peyrin, T.: Improved Differential Attacks for ECHO and Grøstl. In: Rabin, T. (ed.) CRYPTO. LNCS, vol. 6223, pp. 370–392. Springer (2010)
33. 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. LNCS, vol. 5365, pp. 91–103. Springer (2008)
34. Sasaki, Y.: Boomerang Distinguishers on MD4-Based Hash Functions: First Practical Results on Full 5-Pass HAVAL. In: Miri, A., Vaudenay, S. (eds.) SAC. LNCS, Springer (2011), to appear
35. Vielhaber, M.: Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive, Report 2007/413 (2007)
36. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE. LNCS, vol. 1636, pp. 156–170. Springer (1999)
37. Wagner, D.: A Generalized Birthday Problem. In: Yung, M. (ed.) CRYPTO. LNCS, vol. 2442, pp. 288–303. Springer (2002)
38. Wang, G.: Related-Key Rectangle Attack on 43-Round SHACAL-2. In: Dawson, E., Wong, D.S. (eds.) ISPEC. LNCS, vol. 4464, pp. 33–42. Springer (2007)
39. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO. LNCS, vol. 3621, pp. 17–36. Springer (2005)
40. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT. LNCS, vol. 3494, pp. 19–35. Springer (2005)
41. Wang, X., Yu, H.: Non-randomness of 39-step SHA-256. Presented at rump session of EUROCRYPT (2008)
42. Watanabe, D., Hatano, Y., Yamada, T., Kaneko, T.: Higher Order Differential Attack on Step-Reduced Variants of *Luffa* v1. In: Hong, S., Iwata, T. (eds.) FSE. LNCS, vol. 6147, pp. 270–285. Springer (2010)