Analysis of simplified variants of SHA-256*

Krystian Matusiewicz[1], Josef Pieprzyk[1],
Norbert Pramstaller[2], Christian Rechberger[2], Vincent Rijmen[2]
{kmatus,josef}@ics.mq.edu.au,
{norbert.pramstaller,christian.rechberger,
vincent.rijmen}@iaik.tugraz.at,
[1]Centre for Advanced Computing, Algorithms and Cryptography,
Department of Computing, Macquarie University, NSW 2106 Australia
[2]Institute for Applied Information Processing and Communications (IAIK),
Graz University of Technology, Inffeldgasse 16a, A–8010 Graz

Abstract: In this paper we analyse the role of some of the building blocks of SHA-256. We show that the disturbance-correction strategy is applicable to the SHA-256 architecture and we prove that functions Σ, σ are vital for the security of SHA-256 by showing that for a variant without them it is possible to find collisions with complexity 2^{64} hash operations. As a step towards an analysis of the full function, we present the results of our experiments on Hamming weights of expanded messages for different variants of the message expansion and show that there exist low-weight expanded messages for XOR-linearised variants.

C. Wolf, S. Lucks, P.-W. Yau (Eds.): WEWoRC 2005, LNI P-74, pp. 123–134, 2005.

1 Introduction

Recent results on the practical cryptanalysis of many hash functions from the MD family, including MD4, MD5 [WLF+05, WY05] as well as SHA-0 and SHA-1 [BCJ+05, RO05, WYY05b, WYY05a], drew a considerable attention to the security of hash functions and raised some questions about the security of the latest function in this family, namely SHA-256. The first published independent analysis of the members of the SHA-2 family was done by Gilbert and Handschuh [GH03]. They showed that there exists a 9-step local collision with probability 2^{-66}. Later on, this result has been improved by Hawkes, Paddon and Rose [HPR04]. They showed how to increase the probability to 2^{-39} using modular differences.

In this paper we investigate the limits of applying the disturbance-correction strategy that was introduced by Chabaud and Joux [CJ98] to cryptanalyse SHA-0. We demonstrate the

* The work described in this paper has been supported in part by the ARC grants DP0451484 and DP0345366, by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT, and by the Austrian Science Fund (FWF) project P18138.

importance of the S-boxes applied in SHA-256. Throughout this paper we use different linearisation models, namely a linearisation with respect to the XOR-operation (XOR-linear) and a linearisation with respect to modular addition (ADD-linear). We start from the analysis of an ADD-linear variant of SHA-256 and derive a differential characteristic that produces collisions for that linear model. Next, we present a zero-output differential characteristic with probability 2^{-64} for the hash function with the Boolean functions. This proves that the application of the functions Σ_0, Σ_1, σ_0, and σ_1 is crucial for the security of the original hash function, since they are replaced by the identity function in this analysis. In parallel to this work, a different variant of SHA-256 was analysed by Yoshida and Biryukov [YB05].

A better understanding of the impact of these functions on the whole design is the next step in the analysis of SHA-256. While the influence of Σ_0 and Σ_1 on the probability of single correction has been studied well by Hawkes et al. [HPR04], as far as we know, there has been no analysis of the message expansion involving σ_0 and σ_1. In this paper we discuss some properties of the message expansion and present our results of the search for low-weight message differences for various (XOR-linear) variants of the message expansion.

2 Description of SHA-256

SHA-256 [Nat02] is an iterated cryptographic hash function based on a compression function that updates the state of eight 32-bit chaining variables A, …, H according to the values of 16 32-bit words M_0, …, M_{15} of the message. The compression function consists of 64 identical steps presented in Figure 1. The step transformation employs bitwise Boolean functions

$$\begin{aligned} Maj(A,B,C) &= (A \wedge B) \vee (A \wedge C) \vee (B \wedge C) \ , \\ Ch(E,F,G) &= (E \wedge F) \vee (\neg E \wedge G) \ , \end{aligned}$$

and two S-boxes

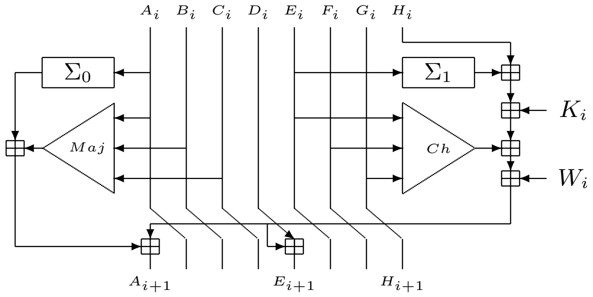
$$\begin{aligned} \Sigma_0(x) &= ROTR^{2}(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x) \ , \\ \Sigma_1(x) &= ROTR^{6}(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x) \ , \end{aligned}$$

built from word rotations to the right ($ROTR$) and bitwise XORs denoted by $\oplus$. The i-th step uses a fixed constant K_i and the i-th word W_i of the expanded message.

The message expansion works as follows. An input message is split into 512-bit message blocks (after padding). A single message block will be denoted either as a row vector $m \in \mathbb{Z}_2^{512}$ or as a vector M of 16 32-bit words $M_t \in \mathbb{Z}_{2^{32}}$, with $0 \leq i < 16$. During the message expansion, this input is expanded into a vector of 64 32-bit words $W_i \in \mathbb{Z}_{2^{32}}$, which may be also seen as the 2048-bit expanded message row-vector w. The words W_i are generated from the initial message M according to the following formula:

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

Figure 1: One step of the SHA-256 compression function



If we set $N = 64$, we get standard SHA-256, taking a different value of N results in a reduced (or extended) variant of it. The functions $\sigma_0(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$ and $\sigma_1(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$ are S-boxes defined using word rotations to the right ($ROTR$) and shifts to the right (SHR).

3 Computing collisions for an ADD-linear variant of SHA-256

In order to analyse the usefulness of a disturbance-correction strategy applied to the SHA-2 architecture, we investigate an ADD-linear variant of SHA-256, where S-boxes are replaced with the identity function,

$$\sigma_0 = \sigma_1 = \Sigma_0 = \Sigma_1 = id \ , \tag{2}$$

and Boolean functions are replaced by the addition modulo 2^{32},

$$Maj(x, y, z) = Ch(x, y, z) = x + y + z \ . \tag{3}$$

Now the whole function consists only of linear operations with respect to the modular addition. If we introduce a difference $\Delta_i = W_i' - W_i$, we can cancel this disturbance by introducing in the next 8 steps $i+1, \ldots, i+8$ the following sequence of corrections

$$\{-4\Delta_i,\ 2\Delta_i,\ 2\Delta_i,\ 4\Delta_i,\ 2\Delta_i,\ \Delta_i,\ 0,\ -\Delta_i\} \ . \tag{4}$$

The whole process of correcting a single disturbance is presented in Table 1. In the first 4 steps we use corrections that keep differences from influencing register A and later from step $i+4$ we successively cancel differences in the register H.

The next step is to find a disturbance pattern Δ that follows the expansion process and can give raise to a corrective pattern. We will use an argument similar to the one used for finding disturbance patterns for SHA-1 [MP05, RO05, PRR05]. Let us introduce the necessary notation first. For any vector $s = [s_0, \ldots, s_l]$, let us denote by $\mathrm{Delay}^a(s)$ a

Table 1: Correcting a single disturbance Δ_i introduced in step i in an ADD-linearised variant of SHA-256

step s	ΔA_s	ΔB_s	ΔC_s	ΔD_s	ΔE_s	ΔF_s	ΔG_s	ΔH_s	ΔW_s
i	0	0	0	0	0	0	0	0	$\boldsymbol{\Delta}_i$
$i+1$	Δ_i	0	0	0	Δ_i	0	0	0	$-4\Delta_i$
$i+2$	0	Δ_i	0	0	$-2\Delta_i$	Δ_i	0	0	$2\Delta_i$
$i+3$	0	0	Δ_i	0	$-\Delta_i$	$-2\Delta_i$	Δ_i	0	$2\Delta_i$
$i+4$	0	0	0	Δ_i	$-\Delta_i$	$-\Delta_i$	$-2\Delta_i$	Δ_i	$4\Delta_i$
$i+5$	0	0	0	0	Δ_i	$-\Delta_i$	$-\Delta_i$	$-2\Delta_i$	$2\Delta_i$
$i+6$	0	0	0	0	0	Δ_i	$-\Delta_i$	$-\Delta_i$	Δ_i
$i+7$	0	0	0	0	0	0	Δ_i	$-\Delta_i$	0
$i+8$	0	0	0	0	0	0	0	Δ_i	$-\Delta_i$
$i+9$	0	0	0	0	0	0	0	0	

vector constructed by preceding elements of s by a zero elements, i.e.

$$\mathrm{Delay}^a(s) = [\underbrace{0,\ldots,0}_{a\text{ times}}, s_0, \ldots, s_l]$$

and by $\mathrm{Delay}^a_n(s)$ the same vector truncated to only n first elements, i.e.

$$\mathrm{Delay}^a_n(s) = [\underbrace{0,\ldots,0}_{a\text{ times}}, s_0, \ldots, s_{n-1-a}] \, .$$

Based on this notation, we can state the following simple fact which will be used later on.

Lemma 3.1 *Let $W \in \mathbb{Z}^{64}_{2^{32}}$. If $\mathrm{Delay}^a(W)$ is a result of the expansion using the recursive formula (1) with $N = 64 + a$, then all the vectors $\mathrm{Delay}^b_{64}(W)$ for $0 \leq b \leq a$ are also results of the expansion process (1).*

PROOF. Each vector $\mathrm{Delay}^b_{64}(W)$ consists of elements of the vector $\mathrm{Delay}^a(W)$ with indices $a-b, a-b+1, \ldots, a-b+63$ and as a part of a sequence following the recurrence relation, also follows the relation. □

The message expansion can be seen as an ADD-linear transformation $E : \mathbb{Z}^{16}_{2^{32}} \to \mathbb{Z}^{64}_{2^{32}}$. This means that E can be written as a 64×16 matrix

$$E = \begin{bmatrix} I_{16} \\ A \\ A^2 \\ A^3 \end{bmatrix}, \tag{5}$$

where I_{16} stands for the identity matrix and A denotes a matrix of the linear transformation producing 16 new words out of 16 old ones according to the recurrence relation (1).

The following theorem fully characterises disturbance patterns for an ADD-linear variant of SHA-256.

Theorem 3.2 *Let $\Delta_M = M' - M$ be a message difference. The expanded difference $\Delta = E(\Delta_M)$ is a valid disturbance vector for an ADD–linear variant of SHA-256 if the following conditions are satisfied:*

$$\mathbf{0} = A^3[8 :: 16] \cdot \Delta_M \ , \tag{6}$$

$$\mathbf{0} = A^{-1}[8 :: 16] \cdot \Delta_M \ , \tag{7}$$

where $M[a :: b]$ means a matrix consisting of rows of the matrix M from the a-th row to the b-th row inclusive.

PROOF. The fundamental observation is that each single word Δ_i of the disturbance vector has to be corrected by adding to the next 8 words the following differences defined by Equation (4),

$$\{-4\Delta_i,\ 2\Delta_i,\ 2\Delta_i,\ 4\Delta_i,\ 2\Delta_i,\ \Delta_i,\ 0,\ -\Delta_i\} \ .$$

This shows that the last non-zero disturbance word may appear in position 55, because we need eight steps 56, ..., 63 to correct it. Thus, the last 8 words of the expanded difference $E \cdot \Delta_M$ have to be zero. Since E is defined by (5), this condition can be written as (6).

Now, let us consider the following linear combination of Δ and its delayed versions

$$\begin{aligned} C = \Delta - 4\,\mathrm{Delay}^1_{64}(\Delta) + 2\,\mathrm{Delay}^2_{64}(\Delta) + 2\,\mathrm{Delay}^3_{64}(\Delta) \\ + 4\,\mathrm{Delay}^4_{64}(\Delta) + 2\,\mathrm{Delay}^5_{64}(\Delta) + \mathrm{Delay}^6_{64}(\Delta) - \mathrm{Delay}^8_{64}(\Delta) \ . \end{aligned} \tag{8}$$

It is easy to see that each disturbance word Δ_i in C is corrected by its appropriate multiplicities appearing in the next eight positions and coming from the delayed vectors. Since the message expansion is linear, C is the result of the expansion if and only if all the delayed and truncated vectors $\mathrm{Delay}^b_{64}(\Delta)$, $0 \le b \le 8$ are results of the expansion process. Lemma 3.1 assures that it is true if $\mathrm{Delay}^8(\Delta) = [0,0,0,0,0,0,0,0,\Delta_0,\ldots,\Delta_{63}]^T$ is the result of the (extended, $N = 68$) expansion process. We can achieve this by taking the first 16 words and expanding them forward according to Equation (1), but also by taking any 16 consecutive words and expanding partly forward and partly backward. In our case we select elements 8–23 for the expansion. If we index elements of $\mathrm{Delay}^8(\Delta)$ starting from -8 and split the vector into two parts: one having negative and the other one having non-negative indices, we can express this requirement equivalently by the following two conditions:

$$[\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,0,0,0,0,0,0,0,0]^T = A^{-1} \cdot \Delta_M \quad \text{and} \quad \Delta = E \cdot \Delta_M .$$

Only the first condition, namely $A^{-1}\Delta_M$ has to end with 8 zeros, has to be satisfied, since Δ is already the result of an expansion. This condition can be written simply in the form of Equation (7) what completes the proof. As long as Equations (6) and (7) are satisfied, Δ is a valid disturbance pattern and C is a complete differential characteristic corresponding to it. □

After obtaining explicit forms of the matrices A^3 and A^{-1} (this is possible since A is a bijection) we solve the system of equations given by (6) and (7) over $\mathbb{Z}_{2^{32}}$ and get the following result:

$$\begin{aligned}\Delta_M = [&\mathtt{0x10000000}, \mathtt{0xa0000000}, \mathtt{0xc0000000}, \mathtt{0xa0000000},\\ &\mathtt{0xe0000000}, \mathtt{0x20000000}, \mathtt{0x40000000}, \mathtt{0x40000000},\\ &\mathtt{0x80000000}, \mathtt{0xd0000000}, \mathtt{0x10000000}, \mathtt{0x60000000},\\ &\mathtt{0x50000000}, \mathtt{0x40000000}, \mathtt{0x70000000}, \mathtt{0x30000000}]^T.\end{aligned} \tag{9}$$

This shows that the solution space is just one-dimensional. Any multiple of Δ_M is also a solution, but since all components of the vector (9) have only up to four most significant binary digits different from zero (so they are all of the form $a_i \cdot 2^{28} \pmod{2^{32}}$ where $a_i \in \{0, \ldots, 15\}$, $0 \leq i < 16$), there are only 16 distinct disturbance patterns. Using any of them results in a collision for ADD-linearised SHA-256.

4 Incorporating Boolean functions

Now let us consider a variant of SHA-256 still without S-Boxes, but with both Boolean functions Maj and Ch in place. If we multiply the basic pattern (9) by 8 (so shift it left by 3 bit positions), we get a disturbance pattern $\Delta^* = E(8\Delta_M)$ that has non-zero bits at the most significant bits only. The most significant bits of Δ^* are as follows

$$\begin{array}{l} 10000000011010111011100110100110 \\ \mathtt{0000011100101111\ \ 1011100000000000}\,. \end{array} \tag{10}$$

Δ^* is a disturbance pattern that not only follows the message expansion but also allows us to treat it as a binary pattern with a relatively low weight of 27.

We can approximate both Boolean functions with probability at least $1/2$ assuming that the function produces an output difference each time the input difference is non-zero. This approximation is shown in Table 2.

Table 2: Probabilities of non-zero output differences for the Boolean functions Ch and Maj

input difference	Ch function		Maj function	
$(\delta_x, \delta_y, \delta_z)$	conditions	prob	conditions	prob
(1,0,0)	$y + z = 1$	1/2	$y + z = 1$	1/2
(0,1,0)	$x = 1$	1/2	$x + z = 1$	1/2
(0,0,1)	$x = 0$	1/2	$x + y = 1$	1/2
(1,1,0)	$x + y + z = 0$	1/2	$x + y = 0$	1/2
(1,0,1)	$x + y = 0$	1/2	$x + z = 0$	1/2
(0,1,1)	–	1	$y + z = 0$	1/2
(1,1,1)	$y + z = 0$	1/2	–	1

If we use this approximation and trace how a single bit disturbance Δ_i^* introduced in step i propagates through the next 8 steps, we get the following sequence of corrections:

$$\{0,\ 0,\ \Delta_i,\ \Delta_i,\ 0,\ 0,\ 0,\ \Delta_i\} \quad , \tag{11}$$

which we need in steps $i+1, \ldots, i+8$ in order to cancel the initial disturbance Δ_i. The whole process is very similar to the one used to obtain the sequence of corrections (as given in (4)).

A complete differential is obtained in the same way as in the previous case, by adding delayed disturbance patterns multiplied by corresponding coefficients of Equation (11), i.e. {0,0,1,1,0,0,0,1}.

This time however, correction process is probabilistic as each active Boolean function almost always (except for input differences $(0,1,1)$ for Ch and $(1,1,1)$ for Maj) introduces a factor of $1/2$. A detailed analysis of these probabilities is presented in Table 3. After multiplication of all factors, we obtain a probability for a successful correction equal to 2^{-84}. Further optimisation are also possible as we can choose messages in such a way that conditions for successful correction will be always satisfied for the first 16 steps, what could increase the probability to around 2^{-64}. This shows that the use of substitution boxes σ_0, σ_1 and Σ_0, Σ_1 is essential for the security of SHA-256 and also demonstrates that mixing only modular additions with Boolean functions is not enough for constructing a secure hash function.

Table 3: Negative exponents e of the probabilities introduced in step s by Boolean functions Maj and Ch. Columns Maj and Ch show input differences to Boolean functions and 2^{-e} gives probabilities introduced by each step.

s	Maj	Ch	e	s	Maj	Ch	e	s	Maj	Ch	e	s	Maj	Ch	e
0	000	000	0	16	110	010	2	32	011	100	2	48	111	110	1
1	100	100	2	17	111	101	1	33	001	010	2	49	111	011	0
2	010	010	2	18	011	010	2	34	000	001	1	50	011	101	2
3	001	101	2	19	101	001	2	35	000	100	1	51	101	010	2
4	000	110	1	20	110	100	2	36	000	010	1	52	110	101	2
5	000	111	1	21	111	110	1	37	000	001	1	53	111	110	1
6	000	011	0	22	011	011	1	38	100	100	2	54	011	011	1
7	000	001	1	23	001	101	2	39	110	110	2	55	001	101	2
8	000	000	0	24	100	110	2	40	111	011	0	56	000	010	1
9	000	000	0	25	110	011	1	41	011	001	2	57	000	101	1
10	100	100	2	26	011	101	2	42	001	100	2	58	000	010	1
11	110	110	2	27	101	110	2	43	100	110	2	59	000	001	1
12	011	111	2	28	010	011	1	44	010	111	2	60	000	000	0
13	101	111	2	29	001	001	2	45	101	011	1	61	000	000	0
14	010	011	1	30	100	000	1	46	110	001	2	62	000	000	0
15	101	101	2	31	110	000	1	47	111	100	1	63	000	000	0

5 The role of S-Boxes

The substitution boxes Σ_0 and Σ_1 constitute the essential part of the hash function and fulfil two tasks: they add bit diffusion and destroy the ADD-linearity of the function. There are modular differentials for Σ_0 and Σ_1 that hold for one bit input difference e with probability 2^{-3} (necessary for S-boxes used in steps $i+1$, $i+5$) and with probability around 2^{-10} for input difference equal to $\Sigma_0(e)$ (used for Σ_1 in step $i+2$). Using the approach of modular differences it is possible to obtain a corrective pattern for the complete round structure with probability around 2^{-42}. A better result of 2^{-39} was obtained by Hawkes *et al.* [HPR04] by explicit computation of modular differences for Σ_0 and Σ_1, rather than approximating them with a constant differential.

The S-boxes σ_0 and σ_1 play a similar role: they provide nonlinearity and better diffusion for the message expansion. These two properties of the message expansion constitute the foundation of the security of the full SHA-256, as in order to apply corrective patterns in a straightforward way, one would need at least 37 expanded words equal to zero (since at most three corrective patterns can be applied). Although this seems to be unlikely, further research is needed in this direction.

In the rest of this section, we concentrate on the message expansion and list some interesting properties of it:

- σ_0 and σ_1 have both the property to increase the Hamming weight of low-weight inputs. This increase is upper bounded by a factor of 3. The average increase of Hamming weight for low-weight inputs is even higher if three rotations are used instead of two rotations and one bit-shift. However, a reason for this bit-shift is given by the next observation.

- In contrast to all other members of the MD4-family including SHA-1, rotating expanded message words to get new expanded message words is not possible anymore (even in the XOR-linearised case). This is due to the bit-shift being used in σ_0 and σ_1.

6 Finding low-weight codewords in the code describing the XOR-linearised SHA-256 message expansion

In the first attempt to get an idea about the effect of all the changes between the SHA-1 message expansion and the SHA-256 message expansion, we consider single bit differences. Table 4 illustrates this comparison. We consider variants reduced to 40 steps as well as full variants (80 steps for SHA-1 variants and 64 steps for SHA-256 variants).

By the modified SHA-1 message expansion we refer to a variant where every XOR is replaced by an addition modulo 2^{32}. By the modified SHA-256 message expansion, we refer to a variant where every addition is replaced by an XOR. We observe that both the introduction of modular additions and the replacement of a single bit-shift by a structure

Table 4: Comparison of the number of affected bits for a single bit difference in various message expansions. In variants using modular addition, we used the all-zero vector as a starting point.

	orig. SHA-1	mod. SHA-1	mod. SHA-256	orig. SHA-256
min (40 steps)	18	18	110	137
max (40 steps)	30	41	297	307
min (full)	107	247	467	507
max (full)	174	354	694	709

using σ_0 and σ_1 heavily increases the number of affected bits in the expanded message.

When talking about the SHA-1 message expansion, it was already observed in the works [MP05, RO05] that weights much smaller than 107 (as given in Table 4) can be found. The minimum weight found for the message expansion of SHA-1 is 44. A more recent treatment of low-weight disturbance patterns in SHA-1 can be found in [JP05].

Due to the nonlinear behaviour of the modular addition, no linear code can describe the SHA-256 message expansion. However, if the modular addition is replaced by XOR, a linear code over $\mathbb{Z}_2$ can be constructed. If we consider SHA-256 with N steps, this code can be represented by a $512 \times 32N$ generator matrix G.

Due to the XOR-linearisation, every possible difference of two expanded words is also a valid word in this code. Therefore, probabilistic algorithms from coding theory [Leo88, Ste89, CC98] can be used to find low-weight differences for the XOR-linearised SHA-256 message expansion. Some results of this codeword search are depicted in Figure 2. All minimum weights found for variants of the message expansion up to the full 64 steps are shown in the figure. Until the 42-step variant, our algorithms found reasonable low weights. This is depicted by the solid line. Considering the 40-step variant, the weight of 26 is low compared to a minimal weight of 110 for single-bit differences given in Table 4. The 40-step expanded message is given in Table 5.

For variants with more than 42 steps, the running time of our algorithms is currently too high to return reasonable low weights. The sudden jump after step 42 is not an intrinsic property of the SHA-256 message expansion, but rather the result of the limited running time of our algorithms.

To show that there indeed are low-weight words for $N > 42$, we proceed as follows. After obtaining a low-weight word for 42 steps we use the expansion process to extend it to the full length word. Weights obtained in this way are depicted by the dashed line. A 42-step word of weight 35 is used there as a starting point. Expanding it to 64 steps gives us a weight of 356. This is considerable lower than 467, which is the minimal weight given for a single bit difference in Table 4. However, there is room for improvements.

In contrast to the words found for the SHA-1 message expansion, there are no zero-bands [RO05] any more. Note that the given expanded message is not necessarily a valid difference in case of the real message expansion since we approximate the modular addition by the bitwise XOR operation. Also note that the given vector cannot directly

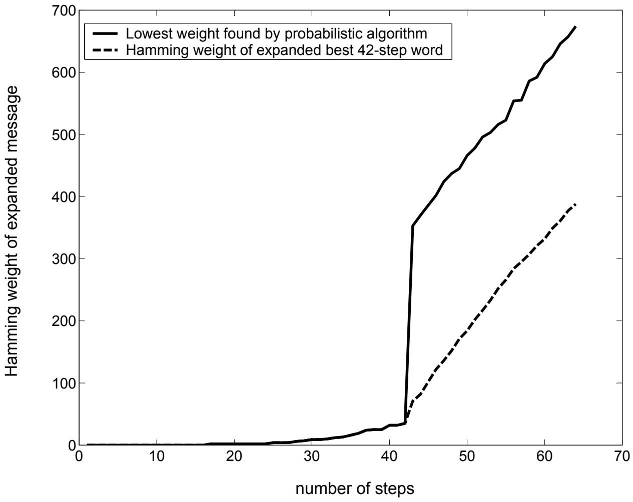


Figure 2: Hamming weights of low-weight words found for step-reduced variants of the XOR-linearised SHA-256 message expansion.

be used as a collision-producing disturbance pattern as described by Chabaud and Joux in their original attack on SHA-0 [CJ98]. The reason is that there are *truncated local collisions* [CJ98] generated by non-zero words in the backward expansion. These local collisions start before step 0 and would cause additional difficulties for constructing a collision-producing differential characteristic. However, we expect to find input words for reduced variants of the message expansion that can be used to build a collision-producing difference.

A number of conditions on chaining variables need to be satisfied in order to ensure that the concatenation of local collisions (which hold with a probability between 2^{-39} and 2^{-42}) results in a collision of the output of the compression function. If we do not assume any pre-fulfilled conditions, the maximal weight we allow for a perturbation pattern is 3 (since $2^{-39 \cdot 4} < 2^{-128}$). Considering the weights in Figure 2, this would mean a maximum of 24 steps.

7 Conclusions

In this paper we presented methods for finding collisions for two simplified variants of SHA-256, one fully linearised with respect to the modular addition and the other one with all the S-Boxes replaced by the identity function. These results show that the presence of S-Boxes is essential for the security of SHA-256. We studied properties of the message

Table 5: Low-weight expanded message for the XOR-linearised 40-step message expansion of SHA-256

00000001	00040088	00000000	00000000
00000000	00000001	00000000	00000000
00000000	15522028	00000000	00000000
00000000	000A0400	00000000	00000000
00000000	00000000	00000000	00000000
00000000	00000000	00004050	00000000
00000000	00000000	00000000	00000000
00000000	00040088	00000001	00000000
00000000	00000001	00000000	00000000
00000001	00000000	00000000	00000000

expansion and presented expanded messages with low Hamming weights for the XOR-linearised message expansion of SHA-256. The general ideas of all these results apply also to other members of the SHA-2 family.

References

[BCJ$^+$05] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and Reduced SHA-1. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT'05*, volume 3494, pages 36–57. Springer, 2005.

[CC98] Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.

[CJ98] Florent Chabaud and Antoine Joux. Differential Collisions in SHA-0. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO'98*, volume 1462 of *LNCS*, pages 56–71. Springer, 1998.

[GH03] Henri Gilbert and Helena Handschuh. Security Analysis of SHA-256 and Sisters. In Mitsuru Matsui and Robert Zuccherato, editors, *Selected Areas in Cryptography, 10th Annual International Workshop, SAC 2003, Ottawa, Canada, August 14-15, 2003, Revised Papers*, volume 3006 of *Lecture Notes in Computer Science*, pages 175–193. Springer, 2003.

[HPR04] Philip Hawkes, Michael Paddon, and Gregory G. Rose. On Corrective Patterns for the SHA-2 Family. Cryptology ePrint Archive, Report 2004/207, August 2004. `http://eprint.iacr.org/`.

[JP05] Charanjit S. Jutla and Anindya C. Patthak. A Matching Lower Bound on the Minimum Weight of SHA-1 Expansion Code. Cryptology ePrint Archive, Report 2005/266, 2005. `http://eprint.iacr.org/`.

[Leo88] Jeffrey S. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Transactions on Information Theory*, 34(5):1354–1359, 1988.

[MP05] Krystian Matusiewicz and Josef Pieprzyk. Finding Good Differential Patterns for Attacks on SHA-1. In *Proc. International Workshop on Coding and Cryptography, WCC'2005*, LNCS, 2005. To appear.

[Nat02] National Institute of Standards and Technology. Secure Hash Standard (SHS). FIPS 180-2, August 2002.

[PRR05] Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Exploiting Coding Theory for Collision Attacks on SHA-1. In Nigel P. Smart, editor, *Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19-21, 2005, Proceedings*, volume 3796 of *LNCS*, pages 78–95. Springer, 2005.

[RO05] Vincent Rijmen and Elisabeth Oswald. Update on SHA-1. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *LNCS*, pages 58–71. Springer, Feb 2005.

[Ste89] Jacques Stern. A method for finding codewords of small weight. In G. Cohen and J. Wolfmann, editors, *Coding Theory and Applications, 3rd International Colloquium*, volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1989.

[WLF$^+$05] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT'05*, volume 3494 of *LNCS*, pages 1–18. Springer, 2005.

[WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT'05*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005.

[WYY05a] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *LNCS*, pages 17–36. Springer, 2005.

[WYY05b] Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. Efficient Collision Search Attacks on SHA-0. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *LNCS*, pages 1–16. Springer, 2005.

[YB05] Hirotaka Yoshida and Alex Biryukov. Analysis of a SHA-256 variant. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography, 12th Annual International Workshop, SAC 2005, Kingston, Ontario, Canada, August 11-12, 2005, Proceedings to appear*, LNCS. Springer, 2005.