Practical Pseudo-Cryptanalysis of Luffa

Jia Keting

Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China

In this paper, some pseudo-collision and pseudo-second-preimage examples are presented for the first round of SHA-3 candidate algorithm Luffa. The pseudo-collision and pseudo-second-preimage can be obtained easily by the message injection function of Luffa. Besides, the pseudo-preimage attacks are shown in this paper. For Luffa-224/256, only 2 iteration computations are needed to get the pseudo-preimage. It is about 2^{64} iteration computations and 2^{64} table lookups to get the pseudo-preimage for Luffa-384 using the extended generalized birthday attack. For Luffa-512, the time complexity is 2^{128} iteration computations and data complexity is 2^{128} table lookups.

Luffa, pseudo-collision, pseudo-second-preimage, pseudo-preimage, generalized birthday attack

1 Introduction

The cryptographic hash function is defined as a function that computes a fixed size message digest from arbitrary size messages, that has been widely used as a fundamental primitive in many cryptographic schemes and protocols, such as electronic signature, authentication of message, electronic commerce and bit commitment, etc. In the past years, the cryptanalysis of hash function has achieved tremendous progress in the construction of collisions. In particular, Wang et al. proposed new techniques to find efficiently collisions on the main hash functions from the MD4 family, e.g. MD4[1], RIPEMD[1], MD5[2], SHA-0 [3] and SHA-1[4]. Moreover the techniques can be applicable to explore the second-preimage of MD4[5], forgery and partial key-recovery attacks on HMAC and NMAC[6]. Responding to advances in the cryptanalysis of hash functions, NIST held two hash workshops to evaluate the security of its approved hash functions and to solicit public comments on its cryptographic hash function policy and standard. Finally, NIST opened a public competition to develop a new hash function called "SHA-3", similar to the development process of the Advanced Encryption Standard (AES). Due to the SHA-3 competition, there are 64 new proposals for hash functions have been submitted to the new SHA-3 algorithm. After the first candidate conference of SHA-3, there are 41 candidate algorithms in the first round. Luffa[7] is one of them, proposed by De Cannière et al.

In this paper, we give some cryptanalytic results of Luffa. The pseudo-collision and pseudo-second-preimage can be obtained easily by the message injection function of Luffa. So this paper shows some pseudo-collision and pseudo-second-preimage examples for Luffa. At the same time, the pseudo-preimage attacks are presented in this paper. For Luffa-224/256, only 2 iteration computations are needed to get the pseudo-preimage. We extend the generalized birthday attack[8] to computer the pseudo-preimage of Luffa-384 with 2^{64} iteration computations and 2^{64} table lookups. The time complexity and data complexity are both 2^{128} to get the pseudo-preimage for Luffa-512.

This paper is organized as follows. In Section 2, we give brief descriptions of Luffa. Section 3 shows some pseudo-collision examples of Luffa. A pseudo-second-preimage example is shown in Section 4. The pseudo-preimage attacks are introduced in Section 5. Finally, we summary our results in Section 6.

2 Preliminaries and Notations

In this section, we first list some notations used

Corresponding author (email: ktjia@mail.sdu.edu.cn)
Supported by the National Natural Science Foundation of China (NSFC Grant No. 60525201) and 973 Project (Grant No.2007CB807902)

in this paper, and then give a brief description of Luffa.

2.1 Notations

$X \| Y$: the concatenation of two messages X and Y.
$h_w(X)$: the high w bits of X.
$l_w(X)$: the low w bits of X.
$\circ$: the multiplication in a ring $GF(2^8)^{32}$.
$\lfloor a \rfloor$: the greatest integer less than or equal to a
$(b_0, b_1, \ldots, b_m)^T$: the transposed matrix of $(b_0, b_1, \ldots, b_m)$, where b_i for $1 \leqslant i \leqslant m$ are column vectors.

2.2 Description of Luffa

Luffa[7], a candidate algorithm for the first round of the SHA-3, was proposed by De Cannière et al. The chaining of Luffa is a variant of a sponge function. Figure 1 depicts the basic structure. The message padding method is to append a single bit '1' followed by the minimum bits of '0' such that the length of the results is a multiple of 256. Let $M = M_0 \| \cdots \| M_{m-1}$ is a message with padding, where $M_i (0 \leqslant i < m)$ are 256-bit blocks. The *iteration function* of Luffa is a composition of a message injection function MI and a permutation P of $w \cdot 256$ bits input. The permutation P includes w permutations Q_0, Q_1, $\ldots$, Q_{w-1}. Let the input of the $i-$th iteration be $(H_0^{(i-1)}, \ldots, H_{w-1}^{(i-1)})$ and M_{i-1}, then the output of the $i-$th iteration is given by

$$\begin{aligned} X_0 \| \cdots \| X_{w-1} &= MI(H_0^{(i-1)}, \ldots, H_{w-1}^{(i-1)}, M(i)), \\ H_j^{(i)} &= Q_j(X_j), j = 0, 1, \ldots, w-1, \end{aligned}$$

where Q_j is a permutation of 256 bits input, $H_j^{(0)} = V_j$, and $V_j (0 \leqslant j < w)$ are the initial values. A finalization is used to the chaining value $(H_0^{(m-1)}, \ldots, H_{w-1}^{(m-1)})$. The finalization consists of iterations of an output function OF and a iteration function with a fixed message $0x00 \cdots 0$, which is called *blank iteration*. The output function OF XORs all block values and outputs the resultant 256-bit value. Let the output at the $i-$th iteration be Z_i, then the output function is defined by

$$Z_i = \bigoplus_{j=0}^{w-1} H_j^{(m-1+i')}.$$

where $i' = i$ if $m = 1$ and $i' = i + 1$ otherwise. The output of Luffa-256 is Z_0, the output of Luffa-512 is $Z_0 \| Z_1$. The outputs of Luffa-224 and Luffa-384 are the truncation of the Luffa-256 and Luffa-512 respectively.

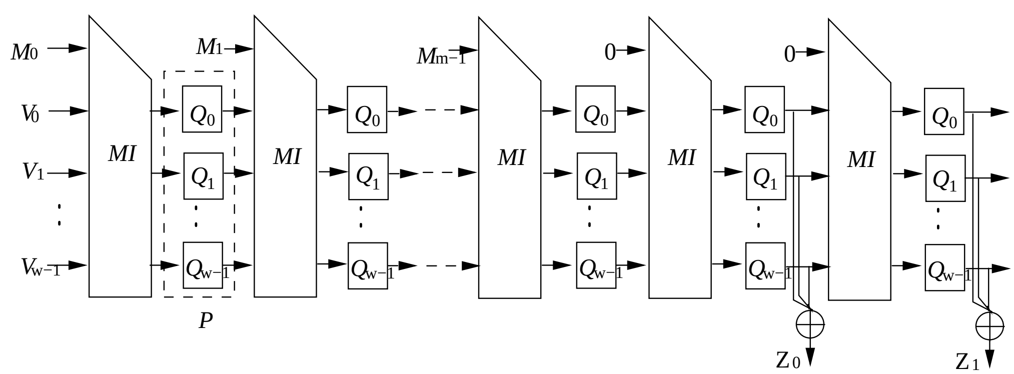

Figure 1 The Structure of Luffa Hash Function.

Message Injection Function MI. The message injection functions MI can be represented by the matrix over a ring $GF(2^8)^{32}$. The definition polynomial of the field is given by $\phi(x) = x^8 + x^4 + x^3 + x + 1$, corresponding to "0x11b". The map from an 8 words value $(h_0, \ldots, h_7)$ to an element of the ring is defined by $(\Sigma_{0 \leqslant k < 8} h_{k,l} x^k)_{0 \leqslant l < 32}$. Let $A_{w \times (w+1)} = (a_0, a_1, \ldots, a_{w-1}, a_w)$ respects the matrix of MI, where $a_i (0 \leqslant i \leqslant w)$ are column vectors. Then $(X_0, X_1, \ldots, X_{w-1})^T = A_{w \times (w+1)} \circ (H_0, H_1, \ldots, H_{w-1}, M)^T$. For Luffa-224/256, the digest is 224 or 256, $w = 3$, and

$$A_{w\times(w+1)} = \begin{pmatrix} 0x3, 0x2, 0x2, 0x1 \\ 0x2, 0x3, 0x2, 0x2 \\ 0x2, 0x2, 0x3, 0x4 \end{pmatrix},$$

where numerics 0x1, 0x2, 0x3, 0x4 correspond to polynomials $1, x, x + 1, x^2$ respectively.

For Luffa-384, the digest is 384, $w = 4$, and

$$A_{w\times(w+1)} = \begin{pmatrix} 0x4, 0x6, 0x6, 0x7, 0x1 \\ 0x7, 0x4, 0x6, 0x6, 0x2 \\ 0x6, 0x7, 0x4, 0x6, 0x4 \\ 0x6, 0x6, 0x7, 0x4, 0x8 \end{pmatrix}.$$

For Luffa-512, the digest is 512, $w = 5$, and

$$A_{w\times(w+1)} = \begin{pmatrix} 0x0F, 0x08, 0x0A, 0x0A, 0x08, 0x01 \\ 0x08, 0x0F, 0x08, 0x0A, 0x0A, 0x02 \\ 0x0A, 0x08, 0x0F, 0x08, 0x0A, 0x04 \\ 0x0A, 0x0A, 0x08, 0x0F, 0x08, 0x08 \\ 0x08, 0x0A, 0x0A, 0x08, 0x0F, 0x10 \end{pmatrix}.$$

3 Pseudo-Collision Attacks on Luffa

Property 1. The rank of the matrix of the message input function is w, for $w = 3, w = 4, w = 5$.

The massage injection function MI is a many-to-one function, the input is $(w + 1) \cdot 256$ bits, but the output is $w \cdot 256$ bits. So, there are 2^{256} inputs for a output. All the 2^{256} inputs with the same output for the iteration function of Luffa, which are pseudo-collisions. Given any $(X_0, X_1, \ldots, X_{w-1})$, we can get 2^{256} inputs. Considering the rank of the matrix is w, we assignment to an elements of the input, Then we can get the remaining elements of the input by the inverse of the matrix. Take $w = 3$ for example. Let $(X_0, X_1, X_2) = (0, 0, 0)$, H_0=(0x9b6a03ec, 0x96c25dd5, 0x9f6fa0ee, 0xeefce4a5, 0xaacd3b44, 0x214bf8b7, 0xc204dd70, 0xa097fadc). Then

$$\begin{pmatrix} H_1 \\ H_2 \\ M \end{pmatrix} = \begin{pmatrix} 0x2, 0x2, 0x1 \\ 0x3, 0x2, 0x2 \\ 0x2, 0x3, 0x4 \end{pmatrix}^{-1} \begin{pmatrix} X_0 + 3H_0 \\ X_1 + 2H_0 \\ X_2 + 2H_0 \end{pmatrix}.$$

Table 1 shows a pseudo-collision for $w = 3$, i.e., $(0, 0, 0) = MI(H_0, H_1, H_2, M_0) = MI(H_0', H_1', H_2', M_0')$. Table 2 shows a pseudo-collision for $w = 4$, i.e., $(0, 0, 0, 0) = MI(H_0, H_1, H_2, H_3, M_0) = MI(H_0', H_1', H_2', H_3', M_0')$. Table 3 shows a pseudo-collision for $w = 5$, i.e., $(0, 0, 0, 0, 0) = MI(H_0, H_1, H_2, H_3, H_4, M_0) = MI(H_0', H_1', H_2', H_3', H_4', M_0')$.

H_0	0x9b6a03ec	0x96c25dd5	0x9f6fa0ee	0xeefce4a5	0xaacd3b44	0x214bf8b7	0xc204dd70	0xa097fadc
H_1	0x47dd7c19	0xc61ae129	0x195aa27f	0x8230a193	0x8f98ffaa	0x5a827d64	0x3c14f9ca	0x7fd7b98a
H_2	0x989d3f4f	0xc5eddd8a	0x49821e83	0xdb45e054	0x3c14f9ca	0x7fd7b98a	0x47dd7c19	0x81c79d30
M_0	0xc5eddd8a	0x8c6fc309	0x0a5ac198	0x7fcc26d1	0x43c34040	0x380ac593	0xc61ae129	0x195aa27f
H_0'	0xde6bedf0	0x219461a1	0x06ebe485	0xf0733600	0x19920b9e	0xfbe0d985	0xc5e0d61c	0x5a06f524
H_1'	0xbc86f3c8	0x2cec6aa1	0x2aea0f07	0xf481e2f7	0x0b7e6ea6	0x4eecf5ba	0x098adff7	0x5c009082
H_2'	0xba80966e	0x4807113f	0x27920407	0xde866cd3	0x098adff7	0x5c009082	0xbc86f3c8	0x906a9969
M_0'	0x4807113f	0x6f951538	0x4394feba	0x6d8c254a	0x558a4f75	0xe086634a	0x2cec6aa1	0x2aea0f07
X_0	0	0	0	0	0	0	0	0
X_1	0	0	0	0	0	0	0	0
X_2	0	0	0	0	0	0	0	0

Table 1 A pseudo-collision for $w = 3$.

H_0	0xde6bedf0	0x219461a1	0x06ebe485	0xf0733600	0x19920b9e	0xfbe0d985	0xc5e0d61c	0x5a06f524
H_1	0x277f8524	0x0f8cba51	0x3eed8eba	0xf46c63d4	0xdc72dd82	0xa1e62ca1	0x1b8b3bec	0xa5f97975
H_2	0xde6bedf0	0x219461a1	0x06ebe485	0xf0733600	0x19920b9e	0xfbe0d985	0xc5e0d61c	0x5a06f524
H_3	0x0f8cba51	0x316134eb	0xedfe684a	0x0f613b72	0x7d94f123	0xba6d174d	0xbe724299	0x8286fc51
M_0	0xd70cb324	0x38066a3f	0xfd0b3d00	0x3cf4bec8	0x5a06f524	0xde6bedf0	0xffff8c51	0xf91468d4
H'_0	0xd1fbeeb0	0xbb2e310f	0x3065e34f	0x25026f3c	0xcce3b4d0	0x89aa250f	0x3227c8eb	0xff59a5f3
H'_1	0x8b4bd240	0x4fd7b083	0x47a86690	0xc67d958c	0xfec47c3b	0x76f380fc	0xe3dc265b	0x958c7a4c
H'_2	0xd1fbeeb0	0xbb2e310f	0x3065e34f	0x25026f3c	0xcce3b4d0	0x89aa250f	0x3227c8eb	0xff59a5f3
H'_3	0x4fd7b083	0x087fd613	0x0a9e215c	0xb3f23bf7	0x8837fcc7	0x952fa6a7	0x76505c17	0x1ec7a80c
M'_0	0xae49bd7c	0x77cd85df	0xb9cfc640	0x6897f41b	0xff59a5f3	0xd1fbeeb0	0x6ad5dfbf	0x5ab03cf0
X_0	0	0	0	0	0	0	0	0
X_1	0	0	0	0	0	0	0	0
X_2	0	0	0	0	0	0	0	0
X_3	0	0	0	0	0	0	0	0

Table 2 A pseudo-collision for $w = 4$.

H_0	0xde6bedf0	0x219461a1	0x06ebe485	0xf0733600	0x19920b9e	0xfbe0d985	0xc5e0d61c	0x5a06f524
H_1	0x6b8a40ec	0x14ea009e	0xd7e13207	0xc8755c3f	0x51789b82	0x9087184a	0xf67553a6	0xa514f856
H_2	0x02f4b151	0x1018d469	0xa7e04907	0x7b929485	0xda74b824	0xc30b3299	0x741e2ed4	0xe66d87cf
H_3	0x418dcec8	0x3a1f5a4d	0xa3129df0	0x48ea901c	0x2aea0f07	0x4807113f	0x27920407	0x6406fabd
H_4	0x558a4f75	0xb50c2c3f	0xcc6a09eb	0x538c2ad3	0xd7e13207	0xa3ff1cd3	0x45929b1c	0x47662a4d
M_0	0x80724d00	0x9d129269	0x96817dec	0x14ea009e	0x57937f07	0xbe9fc3ba	0xce733999	0x581fc556
H'_0	0xd1fbeeb0	0xbb2e310f	0x3065e34f	0x25026f3c	0xcce3b4d0	0x89aa250f	0x3227c8eb	0xff59a5f3
H'_1	0x0b03f894	0x502e1dac	0x526c3608	0x52cfeae3	0xafd464b4	0x7e8c56ef	0xe9420707	0x69a9f138
H'_2	0xd31a19ff	0x4f746c68	0xbab3e8c7	0x447794fc	0x32841400	0x02422ba4	0x0ba0247f	0xa6366b6f
H'_3	0x1c8583a8	0x58f21754	0xa5e99903	0x6337d064	0xeba3f048	0x9f125b10	0x776e5934	0x44d44817
H'_4	0xb08e1570	0x26dd9d50	0xb1b01053	0x7cce7d4b	0x526c3608	0x59cc1277	0xfffa7918	0x2ce060e7
M'_0	0xcdddb1f3	0x1e6474e7	0xb2cc3ed4	0x502e1dac	0x9fb187fb	0x8a75d763	0x9e8f82d8	0xd0663778
X_0	0	0	0	0	0	0	0	0
X_1	0	0	0	0	0	0	0	0
X_2	0	0	0	0	0	0	0	0
X_3	0	0	0	0	0	0	0	0
X_4	0	0	0	0	0	0	0	0

Table 3 A pseudo-collision for $w = 5$.

4 Pseudo-Second-Preimage Attack on Luffa

Given the message $M = M_0 \| M_1$. Firstly, the adversary computers $MI(V_0, V_1, \ldots, V_{w-1}, M_0) = (X_0, X_1, \ldots, X_{w-1})$. Secondly, he can get another message M_0' and initial value $(V_0', V_1', \ldots, V_{w-1}')$ through the inverse of the message injection function. Then the message $M' = M_0' \| M_1$ with the initial value $(V_0', V_1', \ldots, V_{w-1}')$ has the same digest with M. Table 4 shows a pseudo-second-preimage example for the message M_0 = (0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa, 0xaaaaaaaa).

V_0	0x6d251e69	0x44b051e0	0x4eaa6fb4	0xdbf78465	0x6e292011	0x90152df4	0xee058139	0xdef610bb
V_1	0xc3b44b95	0xd9d2f256	0x70eee9a0	0xde099fa3	0x5d9b0557	0x8fc944b3	0xcf1ccf0e	0x746cd581
V_2	0xf7efc89d	0x5dba5781	0x04016ce5	0xad659c05	0x0306194f	0x666d1836	0x24aa230a	0x8b264ae7
M_0	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa
V_0'	0x6d251e68	0x44b051e0	0x4eaa6fb4	0xdbf78465	0x6e292011	0x90152df4	0xee058139	0xdef610bb
V_1'	0xc3b44b94	0xd9d2f256	0x70eee9a0	0xde099fa2	0x5d9b0557	0x8fc944b3	0xcf1ccf0f	0x746cd581
V_2'	0xf7efc89c	0x5dba5781	0x04016ce5	0xad659c04	0x0306194e	0x666d1836	0x24aa230b	0x8b264ae6
M_0'	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaaa	0xaaaaaaab	0xaaaaaaab	0xaaaaaaab	0xaaaaaaaa	0xaaaaaaaa
X_0	0xe6333b1e	0x96d8e9f6	0x24d83129	0x6aa44be3	0x4da482a5	0x0a0bbb57	0x3d1e5ae2	0x71efd72c
X_1	0x48a26ee2	0xa110e0ea	0x1a9cb73d	0xc5f0fa8f	0xd4bc0d49	0x15d7d210	0x1c0714d5	0xdb751216
X_2	0x7cf9edea	0x2578453d	0xc4d998d2	0xb69cf929	0x208bbbfb	0x56d9243f	0xf7b1f8d1	0x243f8d70

Table 4 A pseudo-second preimage for the message M_0 when $w = 3$

5 Pseudo-Preimage Attack on Luffa

First we give our observation in the following, which is used to get the pseudo-preimage of Luffa.

Proposition 1. The blank iteration is a permutation.

Proof. For the blank iteration, the message is 0. The message injection function MI is $(X_0, X_1, \ldots, X_{w-1}) = MI(H_0, H_1, \ldots, H_{w-1}, 0)$. Since the rank of the matrix of MI is w, for $w = 3, w = 4, w = 5$. Thus

$$(H_0, \ldots, H_{w-1})^T = (a_0, \ldots, a_{w-1})^{-1}(X_0, \ldots, X_{w-1})^T.$$

P is a permutation, so the blank iteration is a permutation. $\square$

5.1 Pseudo-Preimage Attack on Luffa-256

For Luffa-256, given a digest Z_0, the adversary can computer a pseudo-preimage with the following steps.

1. Assignment to Y_0, Y_1 with arbitrary value. Without loss of generality, let $Y_0 = Y_1 = 0$. Then $Y_3 = Z_0$.
2. Invert the permutation P. $X_0 = Q_0^{-1}(Y_0)$, $X_1 = Q_1^{-1}(Y_1)$, $X_2 = Q_2^{-1}(Y_2)$.
3. Invert the message injection function MI.

$$\begin{pmatrix} H_0 \\ H_1 \\ H_2 \end{pmatrix} = \begin{pmatrix} 0x3, 0x2, 0x2 \\ 0x2, 0x3, 0x2 \\ 0x2, 0x2, 0x3 \end{pmatrix}^{-1} \begin{pmatrix} X_0 \\ X_1 \\ X_2 \end{pmatrix}.$$

4. The adversary computers the inverse of the permutation P, $X_0 = Q_0^{-1}(H_0)$, $X_1 = Q_1^{-1}(H_1)$, $X_2 = Q_2^{-1}(H_2)$. For (X_0, X_1, X_2), the adversary can obtain the input (V_0', V_1', V_2') and message M_0 by the method mentioned in the Section 3. So the digest of M_0 with initial value (V_0', V_1', V_2') is Z_0.

For Luffa-384, the digest is Z_0 and the high 128 bits of Z_1. $Z_1 = Z_{1,0} \| Z_{1,1} \| Z_{1,2} \| Z_{1,3} \| Z_{1,4} \| Z_{1,5} \| Z_{1,6} \| Z_{1,7}$, where $Z_{1,i}$ for $0 \leqslant i < 8$ are 32-bit words. The adversary randomly chooses (H_0, H_1, H_2), and $H_3 = H_0 \oplus H_1 \oplus H_2 \oplus Z_0$, computer Z_1'. If the $Z_{1,0}' \| Z_{1,1}' \| Z_{1,2}' \| Z_{1,3}'$ = $Z_{1,0} \| Z_{1,1} \| Z_{1,2} \| Z_{1,3}$, let $Y_0 = H_0, Y_1 = H_1, Y_2 = H_2$, the adversary can computer (V_0', V_1', V_2', V_3') and message M_0, which have the same digest $Z_0\|Z_{1,0}\|Z_{1,1}\|Z_{1,2}\|Z_{1,3}$, using Step 2, 3, 4 mentioned above. The complexity is 2^{127} iteration functions. For Luffa-512, the complexity is 2^{255} by the similar attack. In the next subsections, we introduce an algorithm to improve the attack on Luffa-384/512 by the generalized birthday attack, proposed by Wagner[8]. The k-dimensional generalization of the birthday problem is, given k lists L_1, L_2, ..., L_k independently at random from $\{0,1\}^n$, to find k elements $x_i \in L_i$ for $1 \leqslant i \leqslant k$, s.t. $x_1 \oplus x_2 \oplus \cdots \oplus x_k = 0$. Wagner's algorithm[8] builds a binary tree starting from the input lists $L_1, L_2, \ldots, L_k$. The time complexity and data complexity are both $t \cdot 2^{\frac{n}{1+t}}$, where $t = \lfloor \log_2 k \rfloor$.

5.2 Pseudo-Preimage Attack on Luffa-384

For Luffa-384, Let $(H_0, H_1, H_2, H_3, 0)$ be the input of the the last blank iteration function, and (X_0, X_1, X_2, X_3) be the output of MI. The digest is $Z_0\|\bar{Z}_1$, where $\bar{Z}_1 = Z_{1,0}\|Z_{1,1}\|Z_{1,2}\|Z_{1,3}$ Then

$$h_{128}(Q_0(X_0) \oplus Q_1(X_1) \oplus Q_2(X_2) \oplus Q_3(X_3)) = \bar{Z}_1. \quad (1)$$

For the message injection function MI, we can get $(H_0, H_1, H_2, H_3)^T = A_{4\times4}^{-1}(X_0, X_1, X_2, X_3)^T$, where $A_{4\times4}$ is the first 4 column vectors of the matrix $A_{4\times5}$, i.e.,

$$A_{4\times4} = \begin{pmatrix} 0x4, 0x6, 0x6, 0x7 \\ 0x7, 0x4, 0x6, 0x6 \\ 0x6, 0x7, 0x4, 0x6 \\ 0x6, 0x6, 0x7, 0x4 \end{pmatrix}.$$

It's inverse matrix is

$$A_{4\times4}^{-1} = \begin{pmatrix} 0x20, 0x43, 0x84, 0x11 \\ 0x11, 0x20, 0x43, 0x84 \\ 0x84, 0x11, 0x20, 0x43 \\ 0x43, 0x84, 0x11, 0x20 \end{pmatrix}.$$

Since $H_0 \oplus H_1 \oplus H_2 \oplus H_3 = Z_0$, thus

$$X_0 \oplus X_1 \oplus X_2 \oplus X_3 = Z_0', \quad (2)$$

where $Z_0' = 0x3 \circ Z_0$.

Obviously, it's necessary for us to get (X_0, X_1, X_2, X_3) to make the Eq.(1) and Eq.(2) hold together. However, there is only one equation in generalization birthday problem. So we extent Wagner's attack for the special case in the following.

1. The adversary constructs two structures

   $S_0 = \{\, X_0 \mid X_0 \in \{0,1\}^n, l_{192}(X_0) = c_0\}$,
   $S_1 = \{\, X_1 \mid X_1 \in \{0,1\}^n, l_{192}(X_1) = c_0 \oplus l_{192}(Z_0')\}$,

   where c_0 is a 192-bit random constant.

2. The adversary computers $Q_0(X_0)$ for each $X_0 \in S_0$, and constructs a table T_1 with item $(X_0, h_{128}(Q_0(X_0)) \oplus \bar{Z}_1)$. For each $X_1 \in S_1$, he computers $Q_1(X_1)$. If the low 64 bits of $h_{128}Q_1(X_1)$ equal the low 64 bits of the second elements of some item in T_1, insert the item $(X_0, X_1, h_{64}(X_0 \oplus X_1 \oplus Z_0')\|(h_{64}(Q_0(X_0) \oplus Q_1(X_1)) \oplus h_{64}(\bar{Z}_1))$ into table T_2. There are about 2^{64} items in table T_2.

3. The adversary constructs two structures
   $S_2 = \{\, X_2 \mid X_2 \in \{0,1\}^n, l_{192}(X_2) = c_1\}$,
   $S_3 = \{\, X_3 \mid X_3 \in \{0,1\}^n, l_{192}(X_3) = c_1\}$,
   where c_1 is a 192-bit random constant.

4. The adversary computers $Q_2(X_2)$ for each $X_2 \in S_2$, and constructs a table T_3 with item $(X_2, h_{128}(Q_2(X_2)))$. For each $X_3 \in S_3$, he computers $Q_3(X_3)$. If the low 64 bits of $h_{128}(Q_3(X_3))$ equal the low 64 bits of the second elements of some item in T_3, insert the item $(X_2, X_3, h_{64}(X_2 \oplus X_3)\|h_{64}(Q_2(X_2) \oplus Q_3(X_3)))$ into table T_4. There are about 2^{64} items in table T_4.

5. Compare the items of table T_2 with T_4. By the birthday attack, there exist two items (One in table T_2, and one in table T_4), whose last elements are the same. Namely, $X_0 \oplus X_1 \oplus Z_0'$ = $X_2 \oplus X_3$ and $h_{128}(Q_0(X_0) \oplus Q_1(X_1)) \oplus \bar{Z}_1 = h_{128}(Q_2(X_2) \oplus Q_3(X_3))$. So the elements X_0, X_1, X_2, X_3 make the Eq.(1) and Eq.(2) hold at the same time.

6. For (X_0, X_1, X_2, X_3), (H_0, H_1, H_2, H_3) can be computed. Then we applize the similar method shown in the subsection 5.1 to get the pseudo-preimage with 2 iteration computations.

Complexity analysis. There are 2^{64} Q_0, Q_1, Q_2, Q_3 computations and 2^{64} table lookups in the above steps. So the time complexity and data complexity are both 2^{64} to get the pseudo-preimage for Luffa-384.

5.3 Pseudo-Preimage Attack on Luffa-512

For Luffa-512, Let $(H_0, H_1, H_2, H_3, H_4, 0)$ be the input of the the last blank iteration function, and $(X_0, X_1, X_2, X_3, X_4)$ be the output of MI. Then

$$Q_0(X_0) \oplus Q_1(X_1) \oplus Q_2(X_2) \oplus Q_3(X_3) \oplus Q_4(X_4) = Z_1. \tag{3}$$

For the message injection function MI, we can get $(H_0, H_1, H_2, H_3, H_4)^T = A^{-1}_{5\times 5}(X_0, X_1, X_2, X_3, X_4)^T$, where $A_{5\times 5}$ is the first 5 column vectors of the matrix $A_{5\times 6}$, i.e.,

$$A_{5\times 5} = \begin{pmatrix} 0xf, 0x8, 0xa, 0xa, 0x8 \\ 0x8, 0xf, 0x8, 0xa, 0xa \\ 0xa, 0x8, 0xf, 0x8, 0xa \\ 0xa, 0xa, 0x8, 0xf, 0x8 \\ 0x8, 0xa, 0xa, 0x8, 0xf \end{pmatrix}.$$

It's inverse matrix is

$$A^{-1}_{5\times 5} = \begin{pmatrix} 0xc7, 0x8b, 0xf4, 0xf4, 0x8b \\ 0x8b, 0xc7, 0x8b, 0xf4, 0xf4 \\ 0xf4, 0x8b, 0xc7, 0x8b, 0xf4 \\ 0xf4, 0xf4, 0x8b, 0xc7, 0x8b \\ 0x8b, 0xf4, 0xf4, 0x8b, 0xc7 \end{pmatrix}.$$

Since $H_0 \oplus H_1 \oplus H_2 \oplus H_3 \oplus H_4 = Z_0$, thus

$$X_0 \oplus X_1 \oplus X_2 \oplus X_3 \oplus X_4 = Z_0', \tag{4}$$

where $Z_0' = 0xf \circ Z_0$.

We can solve the Eq.(3) and Eq.(4) to get $(X_0, X_1, X_2, X_3, X_4)$, using the similar algorithm mentioned above.

1. The adversary constructs two structures
$$S_0 = \{\, X_0 \mid X_0 \in \{0,1\}^n, l_{128}(X_0) = c_0\},$$
$$S_1 = \{\, X_1 \mid X_1 \in \{0,1\}^n, l_{128}(X_1) = c_0\},$$
where c_0 is a 128-bit random constant.

2. The adversary computers $Q_0(X_0)$ for each $X_0 \in S_0$, and constructs a table T_1 with item $(X_0, Q_0(X_0))$. For each $X_1 \in S_1$, he computers $Q_1(X_1))$. If the low 128 bits of $Q_1(X_1)$ equal the low 128 bits of the second elements of some item in T_1, insert the item $(X_0, X_1, h_{128}(X_0 \oplus X_1) \| h_{128}(Q_0(X_0) + Q_1(X_1)))$ into table T_2. There are about 2^{128} items in table T_2.

3. The adversary constructs two structures
$$S_2 = \{\, X_2 \mid X_2 \in \{0,1\}^n, l_{128}(X_2) = c_1 \oplus l_{128}(Z_0')\},$$
$$S_3 = \{\, (X_3, X_4) \mid X_3, X_4 \in \{0,1\}^n, l_{128}(X_3 \oplus X_4) = c_1\},$$
where c_1 is a 128-bit random constant.

4. The adversary computers $Q_2(X_2)$ for each $X_2 \in S_2$, and constructs a table T_3 with item $(X_2, Q_2(X_2))$. For each $(X_3, X_4) \in S_3$, he computers $Q_3(X_3) \oplus Q_4(X_4) \oplus Z_1$. If its low 128 bits equal the low 128 bits of the second elements of some item in T_3, insert the item $(X_2, X_3, X_4, h_{128}(X_2 \oplus X_3 \oplus X_4 \oplus Z_0') \| h_{128}(Q_2(X_2) \oplus Q_3(X_3) \oplus Q_4(X_4) \oplus Z_1))$ into table T_4. There are about 2^{128} items in table T_4.

5. Compare the items of table T_2 with T_4. By the birthday attack, there exist two items (One in table T_2, and one in table T_4), which have the same last ementents. Namely, $X_0 \oplus X_1 \oplus Z_0'$ = $X_2 \oplus X_3 \oplus X_4$ and $Q_0(X_0) \oplus Q_1(X_1)) = Q_2(X_2) \oplus Q_3(X_3) \oplus Q_4(X_4) \oplus Z_1$. So the elements X_1, X_2, X_3, X_4, X_5 make the Eq.(3) and Eq.(4) hold together.

6. For $(X_0, X_1, X_2, X_3, X_4)$, $(H_0, H_1, H_2, H_3, H_4)$ can be computed. Then we use the similar method shown in the subsection 5.1 to get the pseudo-preimage with 2 iteration computations.

Complexity analysis. There are 2^{128} Q_0, Q_1, Q_2, Q_3 and Q_4 computations and 2^{128} table lookups in the above steps. It's about 2^{128} iteration computations to get the pseudo-preimage for Luffa-512.

6 Conclusion

In this paper, we give the pseudo-collision, pseudo-second-preimage and pseudo-preimage attacks for

Luffa. For arbitrary output of the message injection function MI, it's easy to get inputs by the inverse of MI. So we can get pseduo-collisions and pseudo-second-preimages easily for Luffa using the message injection function MI only. We can get a pseudo-preimage for Luffa-224/256 with 2 iteration computations. We extent the generalized birthday attack to find the pseudo-preimage for Luffa-384 with 2^{64} iteration computations and 2^{64} table lookups. It's about 2^{128} iteration computations and 2^{128} table lookups to find pseudo-preimage for Luffa-512.

Acknowledgements. We would like to thank Wang, Desmedt and Han. Their personal discussion about the ROS2 problem, which is the generalization of ROS problem, motivates us to improved the pseudo-preimage attack on Luffa-384/512.

1 Wang X Y, Lai X J, Feng D G et al. Cryptanalysis of the hash functions MD4 and RIPEMD. In: Cramer R, ed. Eurocrypt 2005, LNCS, Vol 3494. Berlin: Springer-Verlag, 2005. 1–18

2 Wang X Y, Yu H B, How to break MD5 and other hash functions. In: Cramer R, ed. Eurocrypt 2005, LNCS, Vol 3494. Berlin: Springer-Verlag, 2005. 19–35

3 Wang X Y, Yu H B, Yin Y L. Efficient collision search attacks on SHA-0. In: Shoup V, ed. CRYPT 2005, LNCS, Vol 3621. Berlin: Springer-Verlag, 2005. 1–16

4 Wang X Y, Yin Y L, Yu H B. Finding collisions in the full SHA-1. In: Shoup V, ed. CRYPT 2005, LNCS, Vol 3621. Berlin: Springer-Verlag, 2005. 17–36

5 Yu H B, Wang G L, Zhang G Y, et al. The second-preimage attack on MD4. In: Desmedt Y, Wang H X, Mu Y, et al., eds. CANS 2005, LNCS, Vol 3810, Berlin: Springer-Verlag, 2005. 1–12

6 Contini S, Yin Y L. Forgery and partial key-recovery attacks on HMAC and NMAC using hash collisions. In: Lai X J, Chen K F, eds. ASIACRYPT 2006. LNCS, Vol 4284. Berlin: Springer-Verlag, 2006. 37–53

7 Cannière C D, Sato H, Watanabe D. Hash function Luffa. http://csrc.nist.gov/groups/ST/hash/sha-3/Round1/documents/Luffa.zip. 2008

8 Wagner D. A generalized birthday problem. In: Yung M, ed. CRYPTO 2002, LNCS, Vol 2442. Berlin: Springer-Verlag, 2002. 288–304