Higher Order Differential Attack on Step-Reduced Variants of *Luffa* v1

Dai Watanabe[1], Yasuo Hatano[1], Tsuyoshi Yamada[2], and Toshinobu Kaneko[2]

[1] Systems Development Laboratory, Hitachi, Ltd.,
292 Yoshida-cho, Totsuka-ku, Yokohama, 244-0817, Japan
[2] Science University of Tokyo,
2641 Yamazaki, Noda, Chiba, 278-8510, Japan
dai.watanabe.td@hitachi.com

Abstract. In this paper, a higher order differential attack on the hash function *Luffa* v1 is discussed. We confirmed that the algebraic degree of the permutation Q_j which is an important non-linear component of *Luffa* grows slower than an ideal case both by the theoretical and the experimental approaches. According to our estimate, we can construct a distinguisher for step-reduced variants of *Luffa* v1 up to 7 out of 8 steps by using a block message. The attack for 7 steps requires 2^{216} messages. As far as we know, this is the first report which investigates the algebraic property of *Luffa* v1. Besides, this attack does not pose any threat to the security of the full-step of *Luffa* v1 nor *Luffa* v2.

Keywords. Hash function, *Luffa*, Higher order differential attack, Non-randomness

1 Introduction

A cryptographic hash function has a lot of application such as a digital signature and a message authentication code. Recently, several important breakthroughs have been made in the cryptanalysis against hash functions and they imply that most of the currently used standard hash functions are vulnerable against new attacks. In these circumstances, National Institute of Standards and Technology (NIST) decided to organize Cryptographic Hash Algorithm Competition (The SHA-3 competition) [13] and started to call for algorithms.

Luffa [6] is a family of hash functions submitted to the SHA-3 competition and was selected as one of the second round candidates. *Luffa* modified its algorithm at the beginning of the second round and the current algorithm is called *Luffa* v2. Throughout this document, we discuss the algorithm submitted to Round 1 (*Luffa* v1) and denote it *Luffa*. The self-security evaluations in the supporting document for the Round 1 [7] mainly discuss generic attacks and differential cryptanalysis. Besides, analyses based on algebraic approach is not discussed seriously in the document. In this paper, we are going to investigate the algebraic property of step-reduced variants of *Luffa* by a higher order differential attack.

An application of a higher order difference to cryptanalysis was suggested by Lai [11] and Knudsen firstly presented the higher order differential attack to a block cipher [10]. The higher order differential attack is a tool to analyze the algebraic property of the target function, especially its algebraic degree. The application to stream ciphers was proposed by Dinur and Shamir [9] and Aumasson *et al.* proposed a cube tester [1, 2] which intends to detect the non-randomness of the target function. The cube tester has been applied not only to stream ciphers, but also to several hash functions submitted to the SHA-3 competition such as MD6 and Hamsi. Recently, Aumasson and Meier proposed the zero-sum attack which is an application of the higher order differential attack [3].

In this paper, firstly we confirm that the algebraic degree of Q_j grows slower than an ideal case both by the theoretical estimate and the experiments. According to our estimate, we can construct a distinguisher for reduced step *Luffa* up to 7 out of 8 steps by using a block message. The attack for 7 steps requires 2^{216} messages. As far as we know, this is the first report which investigates the algebraic property of *Luffa* v1. Besides, this attack does not pose any threat to the security of the full-step of *Luffa* v1 nor *Luffa* v2.

The rest of this paper is organized as follows: Firstly the specification of *Luffa* is briefly introduced in Section 2. Secondly the definition of the higher order difference and its basic property is introduced in Section 3. The increase of the algebraic degree by the iteration of the step function is investigated in Section 4. Then the higher order differential attack on step-reduced variant of the permutation Q_j and its extension to the hash function is given in Section 5. We conclude the discussion in Section 6.

2 Specification of *Luffa*

In this section, we introduce a part of the specification of *Luffa* which is needed to describe the attack. Please refer to [6] for the detail of the specification.

2.1 Chaining

The chaining of *Luffa* is a variant of a sponge function [4, 5]. Figure 1 shows the basic structure of the chaining. The chaining of a hash function consists of iterations of a round function. The message is padded by `10...0` in order to the padded message length is divisible by 256.

Round Function The round function is a composition of a message injection function MI and w permutations Q_j of 256 bits input (See Figure 1). Let the input of the i-th round be $(H_0^{(i-1)}, \ldots, H_{w-1}^{(i-1)})$, then the output of the i-th round is given by

$$H_j^{(i)} = Q_j(X_j), \quad 0 \le j < w,$$
$$X_0 || \cdots || X_{w-1} = MI(H_0^{(i-1)}, \ldots, H_{w-1}^{(i-1)}, M^{(i)}),$$

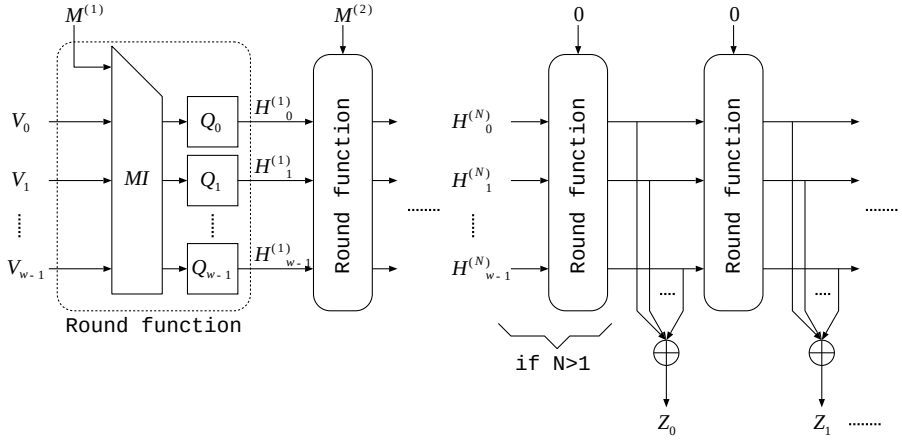

Fig. 1. The *Luffa* construction

where $H_j^{(0)} = V_j$.

In the specification of *Luffa*, the input length of the sub-permutation Q_j is fixed to $n_b = 256$ bits, and the number of the sub-permutations w is 3, 4 and 5 for the hash length 256, 384 and 512 bits respectively.

The message injection functions can be represented by a matrix over the ring $\mathrm{GF}(2^8)^{32}$. The map from an 8 words value $(a_0, \ldots, a_7)$ to an element of the ring is defined by $(\sum_{0 \le k < 8} a_{k,l} x^k)_{0 \le l < 32}$. Note that the least significant word a_7 is the coefficient of the heading term x^7 in the polynomial representation.

Finalization The finalization consists of iterations of an output function OF and a round function with a fixed message `0x00...0`. If the number of (padded) message blocks is more than one, a blank round with a fixed message block `0x00...0` is applied at the beginning of the finalization.

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

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

where $i' = i$ if $N = 1$ and $i' = i + 1$ otherwise. If the hash length is 256-bit, the output is Z_i. For longer hash lengths, more than one round outputs are used to generate the hash values.

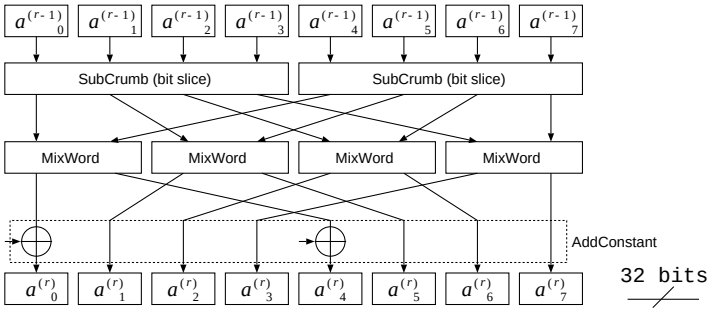

Fig. 2. The step function

2.2 Non-Linear Permutation

The permutation Q_j is defined as a composition of an input tweak and iterations of a step function `Step`. The number of iterations of a step function is 8 and the tweak is applied only once per a permutation.

At the beginning of the step function process, the 256 bits data stored in 8 32-bit registers is denoted by $a_k^{(r)}$ for $0 \leq k < 8$. The data before applying the permutation Q_j is denoted by b_k and the data after the tweak is denoted by $a_k^{(0)}$. The step function consists of the following three functions; `SubCrumb`, `MixWord`, `AddConstant`. The pseudo code for Q_j is given by

```
Permute(a[8], j){ //Permutation Q_j
    Tweak(a);
    for (r = 0; r < 8; r++){
        SubCrumb(a[0],a[1],[2],a[3]);
        SubCrumb(a[4],a[5],[6],a[7]);
        for (k = 0; k < 4; k++)
            MixWord(a[k],a[k+4]);
        AddConstant(a, j, r);
    }
}
```

Each function is described below in turn and the tweaks are described in Section 2.2. We omit the description of `AddConstant` because it is not needed in this paper.

Substitution `SubCrumb` substitutes l-th bits of a_0, a_1, a_2, a_3 (or a_4, a_5, a_6, a_7) by an Sbox S defined by

$$S[16] = \{7, 13, 11, 10, 12, 4, 8, 3, 5, 15, 6, 0, 9, 1, 2, 14\}.$$

Let the output of `SubCrumb` be x_0, x_1, x_2, x_3 (or x_4, x_5, x_6, x_7). Then the substitution by `SubCrumb` is given by

$$x_{3,l}||x_{2,l}||x_{1,l}||x_{0,l} = S[a_{3,l}||a_{2,l}||a_{1,l}||a_{0,l}], \quad 0 \le l < 32,$$
$$x_{7,l}||x_{6,l}||x_{5,l}||x_{4,l} = S[a_{7,l}||a_{6,l}||a_{5,l}||a_{4,l}], \quad 0 \le l < 32.$$

Linear Diffusion `MixWord` is a linear permutation of two words. Let the output words be y_k and y_{k+4} where $0 \le k < 4$. Then `MixWord` is given by the following equations:

$$\begin{aligned} y_{k+4} &= x_{k+4} \oplus x_k, \\ y_k &= x_k \lll \sigma_1, \\ y_k &= y_k \oplus y_{k+4}, \\ y_{k+4} &= y_{k+4} \lll \sigma_2, \\ y_{k+4} &= y_{k+4} \oplus y_k, \\ y_k &= y_k \lll \sigma_3, \\ y_k &= y_k \oplus y_{k+4}, \\ y_{k+4} &= y_{k+4} \lll \sigma_4. \end{aligned}$$

The parameters σ_i are given by $\sigma_1 = 2, \sigma_2 = 14, \sigma_3 = 10, \sigma_4 = 1$.

Tweaks For each permutation Q_j, the least significant four words of a 256-bit input are rotated by j bits to the left in 32-bit registers. Let the j-th block, k-th word input be $b_{j,k}$ and the tweaked word (namely the input to the first step function) be $a^{(0)}_{j,k}$, then the tweak is defined by

$$a^{(0)}_{j,k,l} = b_{j,k,l}, \quad 0 \le k < 4,$$
$$a^{(0)}_{j,k,l} = b_{j,k,(l-j \bmod 32)}, \quad 4 \le k < 8.$$

3 Higher Order Differential Attack

An application of a higher order difference to cryptanalysis was suggested by Lai [11] and Knudsen firstly presented the higher order differential attack to a block cipher [10]. The higher order differential attack is a tool to analyze the algebraic property of the target function, especially its algebraic degree.

In this section, we give a definition of the higher order difference. In addition, the meaning of the distinguishing attack on a hash function is discussed.

3.1 Higher Order Difference

Let $Y = f(X)$ be a function where $X \in \mathrm{GF}(2)^n$, $Y \in \mathrm{GF}(2)^m$. Let $\{A_1, \ldots, A_i\}$ be a set of linearly independent vectors in $\mathrm{GF}(2)^n$ and $V^{(i)}$ be the sub-space

spanned by these vectors. The i-th order difference is defined by

$$\Delta_{V^{(i)}} f(X) = \sum_{A \in V^{(i)}} f(X + A).$$

In the following, $\Delta^{(i)}$ denotes $\Delta_{V^{(i)}}$ if the choice of $V^{(i)}$ does not matter in the discussion. The basic fact of the higher order difference is that $\Delta^{(D+1)} f(X) = 0$ if the algebraic degree of f with respect to X is D. Therefore the higher order difference is used as the tool to evaluate the algebraic degree of the target function.

In addition to the original definition of the higher order difference, we import some terms and notations from the SQUARE attack. The SQUARE attack was proposed by Daemen *et al.* in 1997 as the dedicated attack on the block cipher SQUARE [8].

Let a Λ-set be a set consisting of 16 states such that their values in some crumbs (4-bit inputs to an S-box) are all different (these crumbs are called *active*) and their values are all equal in other crumbs (called *passive*). The basic idea of SQUARE attack is that a permutation preserves the status active or passive. In the higher order differential attack, this observation is useful to choose $V^{(i)}$. If $V^{(i)}$ consists of active crumbs and passive crumbs, the increase of algebraic order at the first Sbox can be ignored by replacing the inputs of the Sboxes by the corresponding outputs.

3.2 Distinguishing Attack on A Hash Function

We will clarify what the terminology *distinguisher* means in this paper.

A distinguisher for a family of functions $\mathcal{F}$ and a set of all functions which maps $\{0,1\}^m$ to $\{0,1\}^n$ is defined as a program that, given a function f, determines if f belongs to $\mathcal{F}$. Therefore, a discussion on distinguishing attack makes sense only if the target function has a parameter. Besides, the naive definition of a collision resistant hash function does not take secret key. Therefore the application of the distinguishing attack in practice is limited to keyed applications such as HMAC. Dealing the IVs as a parameter (as in the discussion of security proof) is another possible situation.

Note that the distinguisher on a hash function (family) only detects a kind of non-randomness property of the target, does not violate collision resistance, second preimage resistance, nor preimage resistance. Even though distinguishing attacks reveal only non-randomness, we believe that this can be a first step to analyze the target function.

By the definition, it is possible to calculate the higher order differences of arbitrary functions including hash functions. Let f be a randomly chosen function whose input length is n-bit. Then the algebraic degree of f is expected $n-1$ so that the event that the i-th order difference $\Delta^{(i)} f$ is rarely zero if i is not much less than n. We use this property as a distinguisher and claim that the attack is successful if such events are detected.

4 Algebraic Degree of Non-linear Permutation Q_j

It is pointed out in [7] that the Boolean polynomial representations of the Sbox of *Luffa* are sparse, especially at the highest degree. The first step of the theoretical estimate is to observe how this property affects the increase of the algebraic degree throughout the iterations of the step functions. In the following, the r iterations of the step function is denoted by $Q_j^{(r)}$. The original permutation of *Luffa* is given by $Q_j = Q_j^{(8)}$.

4.1 Boolean Representations of Sbox

Let the inputs and outputs of the Sbox be $x_{0,l}, x_{1,l}, x_{2,l}, x_{3,l}$ and $y_{0,l}, y_{1,l}, y_{2,l}, y_{3,l}$. Then the polynomial representations of the relations between the input and output bits are given by

$$\begin{aligned}
y_{0,l} &= 1 + x_{2,l} + x_{0,l}x_{1,l} + \underline{x_{1,l}x_{3,l} + x_{2,l}x_{3,l} + x_{0,l}x_{1,l}x_{3,l}},\\
y_{1,l} &= 1 + x_{0,l} + x_{2,l} + x_{0,l}x_{1,l} + x_{0,l}x_{2,l} + x_{3,l} + \underline{x_{1,l}x_{3,l} + x_{2,l}x_{3,l} + x_{0,l}x_{1,l}x_{3,l}},\\
y2,l &= 1 + x_{1,l} + \underline{x_{1,l}x_{3,l} + x_{2,l}x_{3,l} + x_{0,l}x_{1,l}x_{3,l}},\\
y_{3,l} &= x_{0,l} + x_{1,l} + x_{2,l} + x_{0,l}x_{1,l} + x_{1,l}x_{2,l} + x_{0,l}x_{1,l}x_{2,l} + x_{1,l}x_{3,l}.
\end{aligned}$$

4.2 Basic Facts

It is clear from the simple observation of the Boolean representations of the Sbox that the terms whose degrees are more than one and which has monomial $x_{3,l}$ in $y_{0,l}, y_{1,l}, y_{2,l}$ are equal. Let $\eta_l \cdot x_{3,l}$ be the common part in $y_{0,l}, y_{1,l}, y_{2,l}$ and $\xi_{k,l}$ be the remainders (The strict definitions of η_l and $\xi_{k,l}$ are given in Section 4.3). Then the multiplication of $y_{k,l}$ and $y_{k',l}$ for $k \neq k'$ is given by

$$y_{k,l} \cdot y_{k',l} = (\xi_{k,l} + \eta_l x_{3,l})(\xi_{k',l} + \eta_l x_{3,l}) = \xi_{k,l}\xi_{k',l} + (\xi_{k,l} + \xi_{k',l} + 1)\eta_l x_{3,l}. \quad (1)$$

Therefore, we get $\deg y_{k,l} \cdot y_{k',l} < \deg y_{k,l} + \deg y_{k',l}$. This indicates that the designer's estimate of the algebraic degree of $Q_j^{(r)}$ is too optimistic. We should carefully estimate it.

On the other hand, `MixWord()` is the function which sums up $y_{k,l}$ over the subscript l: $z_{k,l} = \texttt{MixWord}(y_k, y_{k+4})_{k,l} = \sum_{\iota \in \Omega_l} y_{k,\iota} + \sum_{\iota \in \Omega'_l} y_{k+4,\iota}$. Then $z_{k,l}$ are given by

$$z_{k,l} = \sum_{\iota \in \Omega_l} \xi_{k,\iota} + \sum_{\iota \in \Omega'_l} \xi_{k+4,\iota} + \sum_{\iota \in \Omega_l} \eta_l x_{3,l} + \sum_{\iota \in \Omega'_l} \eta'_l x_{7,l} \quad (2)$$

for $k = 0, 1, 2$, where η'_l is calculated in the same manner as η_l but differs at the choice of the variables. η'_l uses $x_{4,l}, x_{5,l}, x_{6,l}, x_{7,l}$ instead of $x_{0,l}, x_{1,l}, x_{2,l}, x_{3,l}$. The property, that the higher degree terms of $y_{0,l}, y_{1,l}, y_{2,l}$ are the same, is preserved by `MixWord()`. `AddConstant()` has no influence on this property.

4.3 Recurrence Relations about Algebraic Degree

The observations in Section 4.2 indicates that only `SubCrumb()` contributes to the increase of the algebraic degree. In the following, we identify the iterations of the Sboxes (`SubCrumb()`) as the iterations of the step functions for the simple discussion.

Let us denote the inputs to the l-th Sbox in the r-th step function by $(x_{0,l}^{(r-1)}, x_{1,l}^{(r-1)}, x_{2,l}^{(r-1)}, x_{3,l}^{(r-1)})$ and denote $\eta_l, \xi_{k,l}$ by

$$\begin{aligned}
\eta_l^{(r)} &= \eta_l(x_{0,l}^{(r)}, x_{1,l}^{(r)}, x_{2,l}^{(r)}) = x_{1,l}^{(r)} + x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)},\\
\xi_{0,l}^{(r)} &= \xi_{0,l}(x_{0,l}^{(r)}, x_{1,l}^{(r)}, x_{2,l}^{(r)}) = 1 + x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)},\\
\xi_{1,l}^{(r)} &= \xi_{1,l}(x_{0,l}^{(r)}, x_{1,l}^{(r)}, x_{2,l}^{(r)}) = 1 + x_{0,l}^{(r)} + x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)} + x_{0,l}^{(r)} x_{2,l}^{(r)},\\
\xi_{2,l}^{(r)} &= \xi_{2,l}(x_{0,l}^{(r)}, x_{1,l}^{(r)}, x_{2,l}^{(r)}) = 1 + x_{1,l}^{(r)},\\
\xi_{3,l}^{(r)} &= \xi_{3,l}(x_{0,l}^{(r)}, x_{1,l}^{(r)}, x_{2,l}^{(r)}) = x_{0,l}^{(r)} + x_{1,l}^{(r)} + x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)} + x_{1,l}^{(r)} x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)} x_{2,l}^{(r)}.
\end{aligned}$$

In other words, $\eta_l \cdot x_{3,l}$ denotes the common terms of the polynomial representations and $\xi_{k,l}$ denotes the different terms which do not have the variable $x_{3,l}$. In addition, we denote the terms of degree d in $\eta_l^{(r)}$, $\xi_{k,l}^{(r)}$ by $\eta_{l,d}^{(r)}$, $\xi_{k,l,d}^{(r)}$ respectively.

Now we are going to estimate the algebraic degree of $x_{k,l}^{(r)}$, $\eta_l^{(r)}$, $\xi_{k,l}^{(r)}$ by the recurrence relations. We approximate the relations in order to simplify their representations and Equation 1 is applied once for each variable in the estimation. Let us denote $\delta_l^{(r)} = \deg \eta_l^{(r-1)} + \deg x_{3,l}^{(r-1)}$, $\epsilon_{k,k',l}^{(r)} = \deg \xi_{k,l}^{(r)} + \deg \xi_{k',l}^{(r)}$. Then we have the following relations:

$$\deg \eta_l^{(r)} \sim \max(\epsilon_{0,1,l}^{(r-1)}, \deg\max(\xi_{0,l}^{(r-1)}, \xi_{1,l}^{(r-1)}) + \delta_l^{(r-1)}), \tag{3}$$

$$\deg \xi_{0,l}^{(r)} \sim \deg \eta_l^{(r)}, \tag{4}$$

$$\deg \xi_{1,l}^{(r)} \sim \max(\deg \xi_{1,l}^{(r-1)}, \deg \xi_{2,l}^{(r-1)}) + \max(\deg \xi_{0,l}^{(r-1)}, \delta_l^{(r-1)}), \tag{5}$$

$$\deg \xi_{2,l}^{(r)} \sim \max(\deg \xi_{1,l}^{(r-1)}, \delta_l^{(r-1)}), \tag{6}$$

$$\deg \xi_{3,l,2} = \max(\deg \xi_{0,l}^{(r-1)}, \deg \xi_{2,l}^{(r-1)}) + \max(\deg \xi_{1,l}^{(r-1)}, \delta_l^{(r-1)}), \tag{7}$$

$$\begin{aligned}
\deg \xi_{3,l,3} \sim \max(&\deg \xi_{0,l}^{(r-1)} + \deg \xi_{1,l}^{(r-1)} + \deg \xi_{2,l}^{(r-1)},\\
&\max(\epsilon_{0,1,l}^{(r-1)}, \epsilon_{0,2,l}^{(r-1)}, \epsilon_{1,2,l}^{(r-1)}) + \delta_l^{(r-1)}),
\end{aligned} \tag{8}$$

$$\deg x_{0,l}^{(r)} \sim \max(\epsilon_{0,1,l}^{(r-2)}, \delta_l^{(r-1)}), \tag{9}$$

$$\deg x_{1,l}^{(r)} \sim \max(\epsilon_{0,1,l}^{(r-2)}, \epsilon_{0,2,l}^{(r-2)}, \delta_l^{(r-1)}), \tag{10}$$

$$\deg x_{2,l}^{(r)} \sim \delta_l^{(r-1)}, \tag{11}$$

$$\begin{aligned}
\deg x_{3,l}^{(r)} \sim \max(&\deg \xi_{0,l}^{(r-2)} + \deg \xi_{1,l}^{(r-2)} + \deg \xi_{2,l}^{(r-2)},\\
&\max(\epsilon_{0,1,l}^{(r-2)}, \epsilon_{0,2,l}^{(r-2)}, \epsilon_{1,2,l}^{(r-2)}, \deg x_{3,l}^{(r-2)}) + \delta_l^{(r-2)},\\
&2\deg \xi_{1,l}^{(r-2)} + \deg \xi_{3,l}^{(r-2)}).
\end{aligned} \tag{12}$$

The detailed calculations to get the relations are given in Appendix A.

4.4 Theoretical Estimate of Algebraic Degrees

Table 1 shows the pace of increase of algebraic degrees of variables $x_{k,l}$, $\xi_{k,l}$, η_l from the recurrent relations 2 to 11 and the initial values at $r = 0, 1$. The input/output length of the non-linear permutation $Q_j^{(r)}$ is 256 bits so that the algebraic degrees are at most 256. However we put the estimated degrees as it is, even if it is more than 256, in order to clarify the pace of increase.

Table 1. Pace of increase of algebraic degrees (Theoretical estimate)

r	$x_{0,l}^{(r)}$	$x_{1,l}^{(r)}$	$x_{2,l}^{(r)}$	$x_{3,l}^{(r)}$	$\xi_{0,l}^{(r)}$	$\xi_{1,l}^{(r)}$	$\xi_{2,l}^{(r)}$	$\xi_{3,l}^{(r)}$	$\eta_l^{(r)}$
0	1	1	1	1	2	2	1	2	2
1	3	3	3	3	5	5	3	7	5
2	8	8	8	7	13	13	8	18	13
3	20	20	20	18	33	33	20	46	33
4	51	51	51	46	84	84	51	117	84
5	130	130	130	117	214	214	130	298	214
6	331	331	331	298	545	545	331	759	545
7	843	843	843	759	1,388	1,388	843	1,933	1,388
8	2,147	2,147	2,147	1,933	3,535	3,535	2,147	4,923	3,535

5 Higher Order Differential Attack on *Luffa*

The designers of *Luffa* expected the algebraic degree of the permutation $Q_j^{(r)}$ is given by 3^r [7]. However, as shown in the previous section, the degree increases slower than the ideal case. In addition, the high order part $\eta_l^{(r-1)} \cdot x_{3,l}^{(r-1)}$ of the variables $x_{k,l}^{(r)}$ are common for $k = 0, 1, 2$. We use this property to construct a distinguisher for the permutation $Q_j^{(r)}$. Then we extend the attack to 7-step *Luffa*.

5.1 Theoretical Estimate

Remind the definitions of $\xi_{k,l}$ and η_l that $x_{k,l}^{(r)} = \xi_{k,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}$. The high order part $\eta_l^{(r-1)} \cdot x_{3,l}^{(r-1)}$ of the variables $x_{k,l}^{(r)}$ are common for $k = 0, 1, 2$, so that it can be eliminated by the addition (on the binary field) $x_{k,l}^{(r)} + x_{k',l}^{(r)}$. We propose to use a 32-bit value $x_k^{(r)} + x_{k'}^{(r)}$ as the higher order differential distinguisher. Now it is clear that the important variables in this attack are $\xi_{0,l}^{(r-1)}, \xi_{1,l}^{(r-1)}, \xi_{2,l}^{(r-1)}$, not $x_{k,l}^{(r)}$.

These observations indicate that the number of steps r for which $Q_j^{(r)}$ can be attacked can be estimated by the maximum degree of $\xi_{k,l}^{(r-2)}$. In Table 1, $\max_k \deg \xi_{k,l}^{(5)}$ is 214 so that there is a distinguisher on $Q_j^{(6)}$, which calculates 214-th order difference. This distinguisher for 6 steps does not depend on the choice of the input space $V^{(i)}$.

By the careful choice of the input space $V^{(i)}$, we can extend this distinguisher to 7 steps. There are two known techniques to skip the increase of the algebraic degree by applying `SubCrumb()` in the first step. The first one is to choose the input space $V^{(i)}$ in which the inputs to the Sboxes are active or passive. For example, if the $V^{(i)}$ takes all values in $x_{k,l}$ for $0 \leq k < 8$ and $0 \leq l < t$, we can ignore the effect of `SubCrumb`() at the first step. The second technique is to vary only a bit per an Sbox. This technique is applicable only if the algebraic degree of the target function is small. Let us denote m_1-th bit to m_2-th bit of the variable x by $x[m_1..m_2]$. The distinguisher for 7 steps takes $x_k[0..26]$ for all k as variables. In other words, all possible values of $x_{k,l}$ for $0 \leq k < 8$ and $0 \leq l < 27$ appear once. This distinguisher requires 2^{216} messages. On the other hand, $Q_j^{(8)}$ is not expected to be distinguishable because $\max \deg \xi_{k,l}^{(6)} = 545 > 256$.

5.2 Experimental Inspection

By performing experiments, we check if the theoretical estimates summarized in 1 are reliable. We applied the "a bit per an Sbox" technique which is one of the two techniques to ignore the effect of the first step, as mentioned in the previous section. We did not apply the other technique. If we did, the active Sboxes are relatively sparse, so that it would be possible to skip the `SubCrumb`() in the second step by choosing a good alignment of the active Sboxes. However, our purpose is not to optimize the attack, but to check if the theoretical estimates summarized in Table 1 is reliable so that this kind of "unexpected" skip is not desired. Therefore, we calculated t-th order differences by varying the least significant t bits of the 32-bit variable $x_0^{(0)}$ for $1 \leq t \leq 32$. We calculated each higher order difference for 100 times by randomly generating the initial states.

The experimental results are summarized in Table 2 where the numerical values show the ratio that one of the equations $x_0^{(r)} = x_1^{(r)}$, $x_0^{(r)} = x_2^{(r)}$, $x_1^{(r)} = x_2^{(r)}$, $x_4^{(r)} = x_5^{(r)}$, $x_4^{(r)} = x_6^{(r)}$, $x_5^{(r)} = x_6^{(r)}$ holds, where r means the number of steps. In other words, the values mean the ratio of the distinguishing attack being successful.

Table 3 shows the comparison between the theoretical estimates (See Table 1) and the experimental results (See Table 2).

We calculated the algebraic degree of $Q_j^{(r)}$ from the experimental results by the order. Let t be the lowest number such that the t-th order differential of $x_k^{(r)}$ is equal to zero with probability one. The degree of $Q_j^{(r)}$ is formally estimated at $t-1$. This may cause the contradictions in Table 3 such that the degree of $\xi_{k,l}^{(r-2)}$ is larger than that of $x_{k,l}^{(r-1)}$ for $r = 1, 2$. In other cases, the Table 3 indicates that

Table 2. The success rates of the distinguishing attacks on the permutation $Q_j^{(r)}$ (Experimental results)

	Number of steps				
Order	1	2	3	4	5
1	1.00	0.39	0.00	0.00	0.00
2	1.00	1.00	0.12	0.00	0.00
3	1.00	1.00	0.56	0.00	0.00
4	1.00	1.00	0.93	0.00	0.00
5	1.00	1.00	1.00	0.00	0.00
6	1.00	1.00	1.00	0.01	0.00
7	1.00	1.00	1.00	0.04	0.00
8	1.00	1.00	1.00	0.16	0.00
9	1.00	1.00	1.00	0.45	0.00
10	1.00	1.00	1.00	0.83	0.00
11	1.00	1.00	1.00	0.97	0.00
12	1.00	1.00	1.00	1.00	0.00
13	1.00	1.00	1.00	1.00	0.00
14	1.00	1.00	1.00	1.00	0.00
15	1.00	1.00	1.00	1.00	0.00
16	1.00	1.00	1.00	1.00	0.00
17	1.00	1.00	1.00	1.00	0.01
18	1.00	1.00	1.00	1.00	0.00
19	1.00	1.00	1.00	1.00	0.00
20	1.00	1.00	1.00	1.00	0.00
21	1.00	1.00	1.00	1.00	0.00
22	1.00	1.00	1.00	1.00	0.03
23	1.00	1.00	1.00	1.00	0.04
24	1.00	1.00	1.00	1.00	0.13
25	1.00	1.00	1.00	1.00	0.21
26	1.00	1.00	1.00	1.00	0.38
27	1.00	1.00	1.00	1.00	0.46
28	1.00	1.00	1.00	1.00	0.71
29	1.00	1.00	1.00	1.00	0.83
30	1.00	1.00	1.00	1.00	0.85
31	1.00	1.00	1.00	1.00	0.93
32	1.00	1.00	1.00	1.00	0.99

Table 3. The summary of the algebraic degrees

Number of steps		1	2	3	4	5	6	7	8
Algebraic degree	Theoretical estimate	1	3	8	20	51	130	–	–
($\max_{0\le k\le 2,l} x_{k,l}^{(r-1)}$)	Experimental result	1	1	7	18	–	–	–	–
Distinguisher's degree	Theoretical estimate	–	2	5	13	33	84	214	–
($\max_{0\le k\le 2,l} \xi_{k,l}^{(r-2)}$)	Experimental result	–	2	5	12	≥ 32	–	–	–

the theoretical estimates in Table 1 are very close to the experimental results in Table 2 [1].

5.3 Higher Order Differential Attack on The Hash Function

The higher order differential attack on a hash function does not violate the central three requirements for a hash function, namely collision resistance, second preimage resistance, preimage resistance. On the other hand, the distinguishing attacks are useful to check whether or not the target function has pseudo-randomness which is also required to a hash function. Here we consider the higher order differential attack on the 7-step *Luffa* hash function.

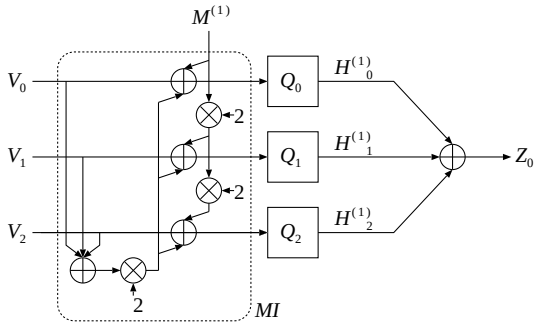

Fig. 3. *Luffa* for a block message ($w = 3$)

The first point of *Luffa* is that there is no blank round if the message length is less than 256 bits. In this case, the message is mixed by the message injection function MI, permuted by non-linear permutation Q_j, then the XORed 256-bit value is output. Therefore, it might be possible to construct a distinguisher based on a higher order difference if the algebraic degree of Q_j is smaller than 256 for all j. Because the only non-linear components in *Luffa* are Q_js which differ only in their tweaks and their step constants. In order to extend the distinguisher for $Q_j^{(7)}$ to the one for the 7-step *Luffa*, we consider the influences by MI and the tweaks. In the following, we show that neither MI nor the tweaks has influence, which is not difficult.

Firstly, the message injection function MI consists of the constant multiplication over $\mathrm{GF}(2^8)^{32}$. This map stabilizes subspaces of $\mathrm{GF}(2^8)^{32}$ given by a natural injection of $\mathrm{GF}(2^8)^t$ where $t \leq 32$. Therefore there is no influence of the message injection function MI if the input space is the direct product of the Λ-set. Secondly, the tweaks rotate the lower 4 words a_4, a_5, a_6, a_7 by j bits to

[1] We append a note that the $t-1$-th order differentials are rarely constants, so that it might be better to estimate the degree of $Q_j^{(r)}$ by t.

the left in a word. Obviously, the tweaks preserve the properties active and passive. Therefore the input space $V^{(i)}$ which cancels the influence of the message injection function also cancels that of tweaks.

These two facts indicate that the distinguisher for $Q_j^{(7)}$ is also applicable to the reduced step hash function as it is.

5.4 Probabilistic Distinguisher

Table 2 shows that the behavior of the distinguisher is probabilistic if the order is less than the expected algebraic degree. Here we discuss how to reduce the complexity of the attack.

If the target function is sufficiently random, the probability to eventually find a local collision $x_k = x_{k'}$ for any k, k' is given by $6 \cdot 2^{-(32+1)/2} \sim 2^{-14}$ and it is small[2]. Therefore $x_k + x_{k'}$ can be used as a distinguisher even if the event is probabilistic. For example, Table 2 shows that 3 of 100 trials successfully found the partial collision with the 22-th order difference for 5 steps. In this case, the computational complexity is $2^{22} \times 100 \sim 2^{28.6}$, which is smaller than the complexity of the attack with the deterministic distinguisher 2^{33}.

On the other hand, we have no idea to theoretically estimate the frequency of this probabilistic event so that it is not clear how much the computational complexity can be reduced in the case of larger number of steps. In addition, the expected degree of the distinguisher for 8 steps is much larger than 256 so that the distinguisher is expected to include many high order terms. We expect that it is difficult to apply the higher order differential distinguisher to 8 steps if the probabilistic event can be observed more often than the deterministic event.

5.5 Zero-sum Attack

Zero-sum attack was recently proposed by Aumasson and Meier [3] and it is an application of higher order differential attack. The basic idea of the zero-sum attack is to choose the Λ-set as the intermediate variables and estimate the increases of the algebraic degrees at the input and output of the target function. If the algebraic degrees (to both sides) are low, there is an certain set of inputs such that (a) their xoring is zero, and (b) the xoring of their corresponding outputs is also zero. This is a property which an *ideal permutation* does not have, and the zero-sum attack uses this property as the distinguisher.

By intuition the zero-sum attack enables to attack double more rounds than the original higher order differential attack. They claimed that the attack on the permutation Q_j of *Luffa* requires 2^{81} inputs. It is obvious that the number of required inputs can be reduced to 2^{33} due to our evaluation result (See Table 1). However, as they mentioned, it is not obvious problem to find an adequate set of messages which satisfies zero-sum property for all Q_j.

[2] In [11] Lai pointed out that $\mathrm{Prob}(\delta_{V_i} f(a) = b)$ is either 0 or at least 2^{i-n} where $f : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$. But this is not our case because the domain of our distinguisher is larger than the range. The probabilistic behavior of our distinguisher may be caused by the terms of high degree of $Q_j^{(r)}$ being sparsely distributed.

5.6 The Higher Order Differential Attack on *Luffa* v2

Luffa changed its algorithm at the beginning of the Round 2 and it is called *Luffa* v2. We do not describe the changes in detail, but the most significant change of *Luffa* v2 in terms of higher order differential attack is that a blank round in the finalization process is applied even if the message length is less than 256 bits. Therefore 16 step functions are always applied for any message block so that their algebraic degree is not likely to be less than 256.

6 Conclusion

In this paper, a higher order differential attack on the hash function *Luffa* is discussed. We confirmed that the algebraic degree of the underlying non-linear permutation Q_j increases slower than expected both by the theoretical estimate and the experiments. According to our estimate, we can construct a distinguisher for reduced step *Luffa* up to 7 out of 8 steps by using a block message. The attack for 7 steps requires 2^{216} messages. As far as we know, this is the first report which investigates the algebraic property of *Luffa* v1. Besides, this attack does not pose any threat to the security of the full-step of *Luffa* v1 nor *Luffa* v2.

7 Acknowledgements

The authors would like to thank Hirotaka Yoshida and the anonymous reviewers of FSE 2010 for their helpful comments and suggestions.

References

1. J.-P. Aumasson, I. Dinur, W. Meier and A. Shamir "Cube Testers and Key Recovery Attacks On Reduced-Round MD6 and Trivium," *Fast Software Encryption, FSE 2009*, Lecture Notes in Computer Science, vol. 5665, Springer-Verlag, pp. 1–22, 2009.
2. J.-P. Aumasson, I. Dinur, L. Henzen, W. Meier, and A. Shamir, "Efficient FPGA Implementations of High-Dimensional Cube Testers on the Stream Cipher Grain-128," *Special-purpose Hardware for Attacking Cryptographic Systems, SHARCS'09*, 2009.
3. J.P. Aumasson and W. Meier, "Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi," 2009. Available at `http://www.131002.net/data/papers/AM09.pdf`.
4. G. Bertoni, J. Daemen, M. Peeters and G. Van Assche, "Sponge Functions," Ecrypt Hash Workshop 2007.
5. G. Bertoni, J. Daemen, M. Peeters and G. Van Assche, "On the Indifferentiability of the Sponge Construction," *Advances in Cryptology, Eurocrypt 2008*, pp. 181–197, 2008.
6. C. De Cannière, H. Sato, D. Watanabe, "Hash Function Luffa: Specification," Submission to NIST SHA-3 Competition, 2008. Available at `http://www.sdl.hitachi.co.jp/crypto/luffa/`.

7. C. De Cannière, H. Sato, D. Watanabe, "Hash Function Luffa: Supporting Document," Submission to NIST SHA-3 Competition, 2008. Available at `http://www.sdl.hitachi.co.jp/crypto/luffa/`.
8. J. Daemen, L. Knudsen, V. Rijmen, "The Block Cipher Square," *Fast Software Encryption, FSE'97*," Lecture Notes in Computer Science, LNCS 1267, Springer-Verlag, pp. 149–165, 1997.
9. I. Dinur and A. Shamir, "Cube Attacks on Tweakable Black Box Polynomials," Cryptology ePrint Archive, Report 2008/385.
10. L. R. Knudsen, "Truncated and Higher Order Differentials," *Fast Software Encryption, FSE '94*, Lecture Note in Computer Science vol. 1008, pp. 196–211, Springer-Verlag, 1994.
11. X. Lai, "Higher order derivatives and differential cryptanalysis," *Proc. Symposium on Communication, Coding and Cryptography*, pp. 227–233, Kluwer Academic Publishers, 1994.
12. National Institute of Standards and Technology, "Secure Hash Standard (SHS)," FIPS 180-2, 2002.
13. National Institute of Standards and Technology, cryptographic hash project, `http://csrc.nist.gov/groups/ST/hash/index.html`.

A Recurrence Relations

The symbol "$\sim$" means the simplification of the expression which (is considered) preserves the algebraic degree.

A.1 Recurrence Relation of $\eta_l^{(r)}$

$$\begin{aligned}\eta_l^{(r)} &= x_{1,l}^{(r)} + x_{2,l}^{(r)} + x_{0,l}^{(r)} x_{1,l}^{(r)} \\ &\sim x_{0,l}^{(r)} x_{1,l}^{(r)} \\ &= (\xi_{0,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)})(\xi_{1,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}) \\ &= \xi_{0,l}^{(r-1)}\xi_{1,l}^{(r-1)} + (\xi_{0,l}^{(r-1)} + \xi_{1,l}^{(r-1)} + 1)\eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &\sim \xi_{0,l}^{(r-1)}\xi_{1,l}^{(r-1)} + (\xi_{0,l}^{(r-1)} + \xi_{1,l}^{(r-1)})\eta_l^{(r-1)} x_{3,l}^{(r-1)}. \end{aligned} \tag{13}$$

A.2 Recurrence Relation of $\xi_{k,l}^{(r)}$

$$\xi_{0,l}^{(r)} = \xi_{0,0}^{(r)} + \xi_{0,1}^{(r)} + \xi_{0,2}^{(r)} \sim \xi_{0,2}^{(r)} = x_{0,l}^{(r)} x_{1,l}^{(r)} = \eta_l^{(r)}. \tag{14}$$

$$\begin{aligned}\xi_{1,l}^{(r)} &= \xi_{1,0}^{(r)} + \xi_{1,1}^{(r)} + \xi_{1,2}^{(r)} \\ &\sim \xi_{1,2}^{(r)} \\ &= x_{0,l}^{(r)} x_{1,l}^{(r)} + x_{0,l}^{(r)} x_{2,l}^{(r)} \\ &= (\xi_{1,l}^{(r-1)} + \xi_{2,l}^{(r-1)})(\xi_{0,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}). \end{aligned} \tag{15}$$

$$\xi_{2,l}^{(r)} = \xi_{2,0}^{(r)} + \xi_{2,1}^{(r)} \sim \xi_{2,1}^{(r)} = x_{1,l}^{(r)} = \xi_{1,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}. \tag{16}$$

$$\begin{aligned} \xi_{3,l,2}^{(r)} &= (x_{0,l}^{(r)} + x_{2,l}^{(r)}) x_{1,l}^{(r)} \\ &= (\xi_{0,l}^{(r-1)} + \xi_{2,l}^{(r-1)})(\xi_{1,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}). \end{aligned} \tag{17}$$

$$\begin{aligned} \xi_{3,l,3}^{(r)} &= x_{0,l}^{(r)} x_{1,l}^{(r)} x_{2,l}^{(r)} \\ &= (\xi_{0,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)})(\xi_{1,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)})(\xi_{2,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}) \\ &\sim \xi_{0,l}^{(r-1)} \xi_{1,l}^{(r-1)} \xi_{2,l}^{(r-1)} \\ &\quad + (\xi_{0,l}^{(r-1)} \xi_{1,l}^{(r-1)} + \xi_{0,l}^{(r-1)} \xi_{2,l}^{(r-1)} + \xi_{1,l}^{(r-1)} \xi_{2,l}^{(r-1)}) \eta_l^{(r-1)} x_{3,l}^{(r-1)}. \end{aligned} \tag{18}$$

A.3 Recurrence Relation of $x_{k,l}^{(r)}$

$$\begin{aligned} x_{0,l}^{(r)} &= \xi_{0,0}^{(r-1)} + \xi_{0,1}^{(r-1)} + \xi_{0,2}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &\sim \xi_{0,2}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &= x_{0,l}^{(r-1)} x_{1,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &= (\xi_{0,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)})(\xi_{1,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)}) + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &\sim \xi_{0,l}^{(r-2)} \xi_{1,l}^{(r-2)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)}. \end{aligned} \tag{19}$$

$$\begin{aligned} x_{1,l}^{(r)} &= \xi_{1,0}^{(r-1)} + \xi_{1,1}^{(r-1)} + \xi_{1,2}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &\sim \xi_{1,2}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &= x_{0,l}^{(r-1)} x_{1,l}^{(r-1)} + x_{0,l}^{(r-1)} x_{2,l}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \\ &\sim \xi_{0,l}^{(r-2)} (\xi_{1,l}^{(r-2)} + \xi_{2,l}^{(r-2)}) + \eta_l^{(r-1)} x_{3,l}^{(r-1)}. \end{aligned} \tag{20}$$

$$x_{2,l}^{(r)} = \xi_{2,0}^{(r-1)} + \xi_{2,1}^{(r-1)} + \eta_l^{(r-1)} x_{3,l}^{(r-1)} \sim \eta_l^{(r-1)} x_{3,l}^{(r-1)}. \tag{21}$$

$$
\begin{aligned}
x_{3,l}^{(r)} &= \xi_{3,1}^{(r-1)} + \xi_{3,l,2}^{(r-1)} + \xi_{3,l,3}^{(r-1)} + x_1^{(r-1)} x_{3,l}^{(r-1)} \\
&\sim \xi_{3,l,3}^{(r-1)} + x_{1,l}^{(r-1)} x_{3,l}^{(r-1)} \\
&= x_{0,l}^{(r-1)} x_{1,l}^{(r-1)} x_{2,l}^{(r-1)} + x_{1,l}^{(r-1)} x_{3,l}^{(r-1)} \\
&= (\xi_{0,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)})(\xi_{1,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)})(\xi_{2,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)}) \\
&\quad + (\xi_{1,l}^{(r-2)} + \eta_l^{(r-2)} x_{3,l}^{(r-2)})(\xi_{3,l,2}^{(r-2)} + \xi_{3,l,3}^{(r-2)} + x_{1,l}^{(r-2)} x_{3,l}^{(r-2)}) \\
&\sim \xi_{0,l}^{(r-2)} \xi_{1,l}^{(r-2)} \xi_{2,l}^{(r-2)} \\
&\quad + (\xi_{0,l}^{(r-2)} \xi_{1,l}^{(r-2)} + \xi_{0,l}^{(r-2)} \xi_{2,l}^{(r-2)} + \xi_{1,l}^{(r-2)} \xi_{2,l}^{(r-2)} + \xi_{3,l,3}^{(r-2)}) \eta_l^{(r-2)} x_{3,l}^{(r-2)} \\
&\quad \xi_{1,l}^{(r-2)} (\xi_{3,l,3}^{(r-2)} + x_{1,l}^{(r-2)} x_{3,l}^{(r-2)}).
\end{aligned} \tag{22}
$$