Improved Integral Attacks on Reduced Round Camellia

YAN-JUN LI[1,2] WEN-LING WU[1] LI-TING ZHANG[1] AND LEI ZHANG[1]

[1]*State Key Laboratory of Information Security, Institute of Software,*
Chinese Academy of Sciences, Beijing 100190, P.R. China
Graduate University of Chinese Academy of Sciences, Beijing 100049, P.R. China
[2]*Beijing Electronic Science and Technology Institute, Beijing 100070, P.R. China*

Abstract. In this paper a method is presented to extend the length of integral distinguisher of Feistel-SP structure, based on which a new 8-round distinguisher of Camellia is proposed. Moreover, we improve integral attacks on reduced round Camellia without *FL/FL*$^{-1}$. We attack 11-round Camellia-128 with the data complexity of 2^{120} and the time complexity of $2^{125.5}$, and 12-round Camellia-256 with the data complexity of 2^{120} and the time complexity of $2^{214.3}$. The result is the best one of integral attacks on reduced round Camellia so far.

Key words. Block cipher, Distinguisher, Integral attack, Camellia, Partial sum technique

1 Introduction

The block cipher Camellia was proposed by NTT and Mitsubishi in 2000[1]. It is based on Feistel structure with SP-type F function and *FL/FL*$^{-1}$ functions layers, and it supports the block length of 128 bits and a variable key length of 128/192/256 bits. Camellia has been accepted by ISO/IEC as an international standard. It is also a winner of NESSIE, CRYPTREC project and IETF. The security of Camellia was initially analyzed by the algorithm designers. Efficient attacks on Camellia include linear cryptanalysis[14], differential cryptanalysis[13,14], impossible differential cryptanalysis[17], truncated differential cryptanalysis [7, 9], higher order differential cryptanalysis [5], collision attack [16] and Square attack [19, 3]. The best attacks on Camellia without *FL/FL*$^{-1}$ function layer were impossible differential cryptanalysis[18], which can attack 12-round Camellia-128 and 16- round Camellia-256 without *FL/FL*$^{-1}$.

Integral attack was extended from Square attack, which is one of the best attacks on AES [2]. Ferguson *et al.* in [4] improved this attack to 8 rounds version of Rijndael-128 with the partial sum technique and the herd technique. Knudsen and Wagner first proposed the definition of integral and analyzed it as a dual to differential attacks particularly applicable to block ciphers with bijective components [8]. Several years later, Reza Z'aba *et al.* presented bit-pattern based integral attack [12]. The integral attack applied to many kinds of block ciphers so far, such as Rijndael [11], ARIA [10], and Serpent [12]. Higher order differential attack and Square attack are different from integral attack, however, their distinguisher length can be extended by using the integral property. In this paper a method is presented to extend the length of distinguisher of Camellia, based on which the effect of integral attack will be improved. Moreover, this method also can be used even to any Feistel-SP structure. Then a new 8-round distinguisher of Camellia without *FL/FL*$^{-1}$ is proposed. Finally, we attack 11-round Camellia-128 with the data complexity of 2^{120} and the time complexity of $2^{125.5}$, and 12-round Camellia-256 with the data complexity of 2^{120} and the time complexity of $2^{214.3}$. The result is the best one of integral attacks on reduced round Camellia so far.

This paper is organized as follows: Section 2 provides a brief description of preliminaries. Section 3 describes a method to extend the length of distinguisher. Section 4 describes the attacks on 11/12-round Camellia. Finally, Section 5 concludes this paper.

2 Preliminaries

2.1 Description of Camellia

Camellia is a Feistel-SP style block cipher with FL/FL^{-1} layers, and the number of rounds are 18/24/24 corresponding to key length of 128/192/256 bits. Additionally, FL/FL^{-1} function is inserted every 6 rounds (Fig.1). The round keys are derived from the master key by means of key scheduling (Fig.2), and the key schedule constants are listed in Table.1. In this paper the input and output of round function are treated as two 8-byte vectors over $F_{2^8}^8$.

Table 1. The key schedule constants

$\sum_1$	0xA09E667F3BCC908B	$\sum_4$	0x54FF53A5F1D36F1C
$\sum_2$	0xB67AE8584CAA73B2	$\sum_5$	0x10E527FADE682D1D
$\sum_3$	0xC6EF372FE94F82BE	$\sum_6$	0xB05688C2B3E6C1FD

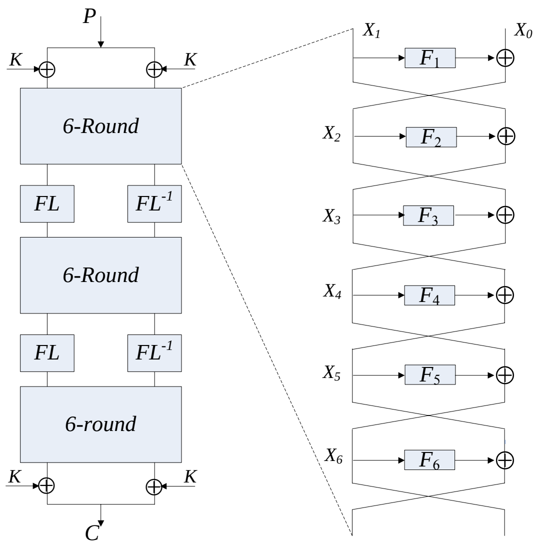


Fig. 1. The Structure of Camellia-128

The round function of Camellia includes three basic operations: Round Key Addition, Substitution Layer and Diffusion Layer (Fig.3). These three basic operations are defined as follows:

Round Key Addition (RKA): The 64-bit round key is Xored to the state.
Substitution Layer (SL): A non-linear byte substitution operation is applied to each byte of the state independently. In Camellia this is implemented by 4 S-boxes with the relationship as follows.

$$s_2(a) = s_1(a) <<< 1; s_3(a) = s_1(a) >>> 1; s_4(a) = s_1(a <<< 1)$$

Diffusion Layer (DL): The diffusion layer is a function $P: F_{2^8}^8 \to F_{2^8}^8$, which is given by

$$\begin{aligned}
m_1 &= x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7 \oplus x_8 & m_5 &= x_1 \oplus x_2 \oplus x_6 \oplus x_7 \oplus x_8 \\
m_2 &= x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7 \oplus x_8 & m_6 &= x_2 \oplus x_3 \oplus x_5 \oplus x_7 \oplus x_8 \\
m_3 &= x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_8 & m_7 &= x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_8 \\
m_4 &= x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & m_8 &= x_1 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7
\end{aligned}$$

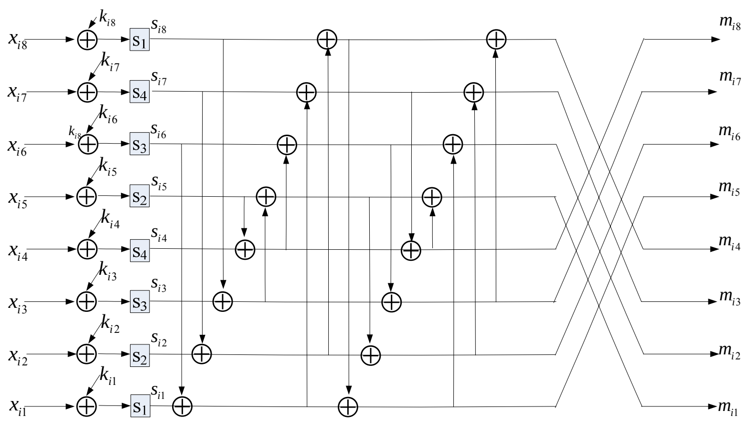


Fig. 2. The Round Function of Camellia

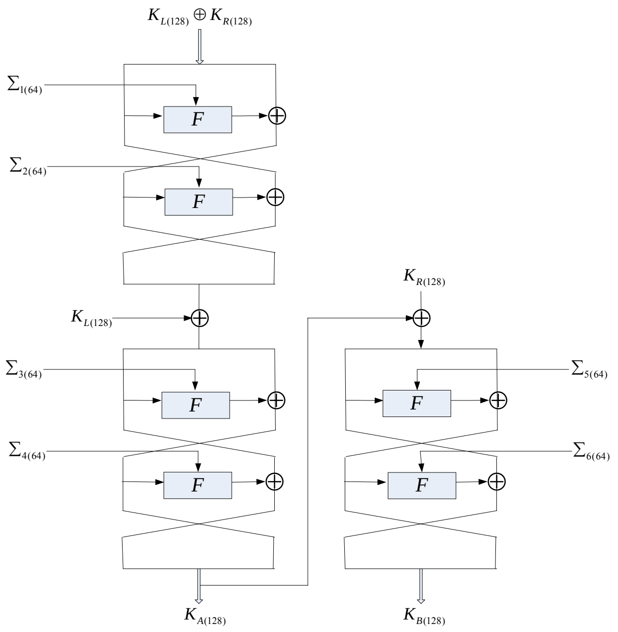


Fig. 3. The Key Schedule of Camellia

2.2 Notations

In the following, we introduce some notations used throughout this paper. The plaintext are denoted as $Plantext = (X_1, X_0)$, where $X_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,8}), i = 0, \ldots, r-1$. Other notations that will be used in this paper are described as follows:

B_r : the output of RKA in r-*th* round.
O_r : the output of SL in r-*th* round.
M_r : the output of DL in r-*th* round.
K_r : the subkey of the r-*th* round.
$b_{r,i}$: the $i+1$-*th* byte of B_r.
$o_{r,i}$: the $i+1$-*th* byte of O_r.
$m_{r,i}$: the $i+1$-*th* byte of M_r.

$k_{r,i}$: the $i+1$-*th* byte of K_r.

λ_i : the active bytes.

2.3 Integral Attack and the Partial Sum Technique

Integral Attack. The integral attack has many interesting features. It can saturate S-Box Layer, and Round Key Addition Layer will not affect this property of saturation. However, the linear transformation influences the length of the integral distinguisher. Integral attack considers a particular collection of *m* bytes in the plaintexts and ciphertexts. The aim of this attack is to predict the values in the sums (i.e. the integral) of the chosen bytes after a certain number of rounds of encryption. In [8], Knudsen and Wagner also generalized this approach to *higher order integrals*: the original set to consider becomes a set of m^d vectors which differ in *d* components and where the sum of this set is predictable after a certain number of rounds. The sum of this set is called a d^{th}*-order integral.* In this paper we not only pay attention to the sum, but also to the appearing times of the sum value.

The Partial Sum Technique. In our attack we will use the partial sum technique. For a value $c_0, c_1, c_2, \ldots, c_l$, we define

$$x_u := \sum_{j=0}^{u} S_j[c_j \oplus k_j].$$

Guessing the values of k_0 and k_1 ,we will complete the transformation $(c_0, c_1, c_2, \ldots, c_l) \rightarrow (x_1, c_2, \ldots, c_l)$. Guessing the values of k_i, we will complete the transformation $(x_{i-1}, c_i, c_{i+1}, \ldots, c_l) \rightarrow (x_i, c_{i+1}, \ldots, c_l)$. In order to obtain the value of x_l, there should be processed *l*-1 steps. If c_i s are in byte pattern, the time complexity of the count of x_l is $2^{8l} \times 2^{16} \times (l-1)$ times S-box lookups. For the details of the complexities of each step, the readers can refer to [4].

3 Integral Distinguishers Based on Feistel-SP Structure

In this section we first explain how to construct a 2nd-order 5-round integral distinguishers (Sec. 3.1), then introduce a method to extend the length of integral distinguishers and the proof also given in detail (Sec. 3.2).

3.1 The 2nd-Order 5-Round Integral Distinguisher

The idea of constructing a 2nd-order 5-round integral distinguisher is like that of constructing 5-round higher order differential distinguishers in Fig.4 [5]. Now we will give different explain as follows.

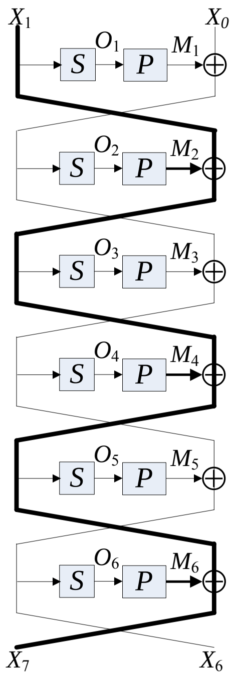


Fig. 4. The 5-Round Higher Order differential Distinguisher

$$X_1 \oplus M_2 \oplus M_4 \oplus M_6 = X_7 \Leftrightarrow X_1 \oplus X_7 = M_2 \oplus M_4 \oplus M_6$$
$$\Leftrightarrow P^{-1}(X_1 \oplus X_7) = O_2 \oplus O_4 \oplus O_6$$

For the 6-th byte, the above equation is equate to

$$[P^{-1}(X_1)]_6 \oplus [P^{-1}(X_7)]_6 = o_{2,6} \oplus o_{4,6} \oplus o_{6,6}, \qquad (1)$$

where $[P^{-1}(X_1)]_6$ is the 6th byte of $P^{-1}(X_1)$.

According to the above description, we will obtain the following result.

Lemma 1. Let the bytes of $x_{0,1}, x_{0,2}$ are active, and other bytes of X_0, X_1 are constants, each value of *t* will appear even times.

$$t = s_3(x_{6,6} \oplus k_{6,6}) \oplus [P^{-1}(X_7)]_6 .$$

Proof. According to Equ.(1), $s_3(x_{6,6} \oplus k_{6,6}) \oplus [P^{-1}(X_7)]_6 = [P^{-1}(X_1)]_6 \oplus o_{2,6} \oplus o_{4,6}$, where $[P^{-1}(X_1)]_6$ $\oplus s_{2,6}$ is constant, so the value of $o_{4,6}$ is important. We find

$$o_{4,6} = s_3(s_1(o_{2,1} \oplus c_1) \oplus s_2(o_{2,1} \oplus o_{2,2} \oplus c_2)$$
$$\oplus s_2(o_{2,1} \oplus o_{2,2} \oplus c_3) \oplus s_3(o_{2,1} \oplus o_{2,2} \oplus c_4) \oplus c_5).$$

Let $o_{2,1} \oplus o_{2,2} = y$, for each value of y, $o_{2,1}$ is active, so each value of $o_{4,6}$ will appear 256 times. Then each value of *t* will appear 256 times. □

3.2 A Method to Extend the Length of Integral Distinguisher

In the structure of Feistel-SP the Xor operation and the permutation P are linear transformations (Fig.5), which can influence the general integral property and also can be used to extend the integral distinguisher. Let some bytes of X_0 be active, and the bytes of X_1 be constant, and then the input of a known t*-round* integral distinguisher is (X_1, X_0). Now we extend it forward one round by the following formula:

$$X_1 = F(X_0) \oplus X_{-1} = P \circ S(K(X_0)) \oplus X_{-1} = P[S(K(X_0)) \oplus P^{-1}(X_{-1})] .$$

Choose X_{-1}, which satisfying that the bytes of $P^{-1}(X_{-1})$ corresponding to the no-constant bytes of $S(K(X_0))$ are active and other bytes are constants. Such (X_0, X_{-1}) will lead to several sets of (X_1, X_0) after one round encryption, and then we will obtain a $t+1$-*round* integral distinguisher. The related proof will be presented in Lemma 2.

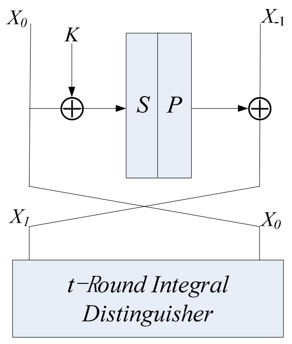


Fig. 5. The Feistel-SP Structure

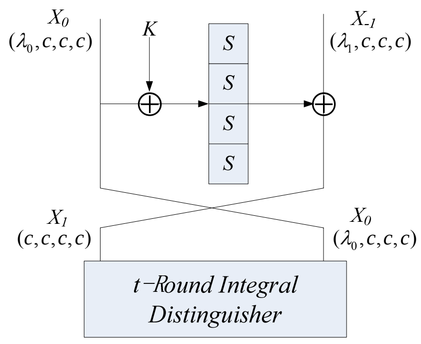


Fig. 6. Example 1

Example 1. In Fig.6 P is an identical transformation. The input of (X_0, X_{-1}) will lead to a $t+1$-*round* integral distinguisher. The λ_0 and λ_1 are two active bytes, so there are 2^{16} values of (X_0, X_{-1}), which will lead to 2^8 sets. In each set $x_{0,1}$ denoted as λ_0 is active and other bytes of (X_1, X_0) are constants denoted as *c*.

Usually the *P* isn't identical transformation, so the choice of X_{-1} will be more complex than described in Example 1. Take Camellia for example, the input of a 2*nd* order 5-round distinguisher is (X_1, X_0), where the bytes of $x_{0,1}, x_{0,2}$ are active and other bytes are constants. Extending three more rounds forward we will obtain

$$X_{-1} = P(\lambda_2, \lambda_3, c, c, c, c, c, c),$$
$$X_{-2} = P(\lambda_4, \lambda_5, \lambda_6, \lambda_7, \lambda_8, \lambda_9, c, \lambda_{10}),$$
$$X_{-3} = P(\lambda_{11}, \lambda_{12}, \lambda_{13}, \lambda_{14}, \lambda_{15}, \lambda_{16}, \lambda_{17}, \lambda_{18}).$$

The $\lambda_i s, 0 \le i \le 18$ are active bytes. Because all bytes of X_{-3} are active, we can't improve it one more round. Then the following lemma is obtained.

Lemma 2. Let the 7th byte of $P^{-1}(X_{-2})$ be constant and other 15 bytes of $(P^{-1}(X_{-2}), X_3)$ be active. After 3 rounds iterative encryption we will obtain 2^{104} sets and in each set the bytes of $x_{0,1}, x_{0,2}$ are active and other bytes are constants.

Proof. We will proof this lemma in three steps as follows:

1) The 2^{32} values of (X_1, X_{-1}) will lead to 2^{16} sets of (X_1, X_0) after one round encryption, in each set the first two bytes of X_1 are active and other bytes are constant.
   The proof of this step is just like that of example 1. We will not repeat it here.
2) The 2^{72} values of (X_{-1}, X_{-2}) will lead to 2^{40} sets of (X_1, X_{-1}) after one round encryption. In each set $x_{1,0}, x_{1,1}$ and the first two bytes of $P^{-1}(X_{-1})$ are active, and other bytes are constants.
   Let all active bytes of $(P^{-1}(X_{-1}), P^{-1}(X_{-2}))$ be denoted as $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9$, and all

constants be denoted as 0. Let the active bytes of $(X_1, P^{-1}(X_{-1}))$ be denoted as y_3, y_4, y_1, y_2. One round encryption will be decrypted as the following equations.

$$\begin{cases} x_1 = y_1 \\ x_2 = y_2 \\ s_1(x_1 \oplus k_1) \oplus x_3 = y_4 \\ s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 = y_3 \\ s_3(x_1 \oplus x_2 \oplus k_3) \oplus x_5 = y_3 \oplus y_4 \\ s_4(x_2 \oplus k_4) \oplus x_6 = y_3 \oplus y_4 \\ s_2(x_1 \oplus x_2 \oplus k_5) \oplus x_7 = y_3 \oplus y_4 \\ s_3(x_2 \oplus k_6) \oplus x_8 = y_4 \\ s_1(x_1 \oplus k_8) \oplus x_9 = y_3 \end{cases}$$

Simplified the above equations, we will obtain the equivalent equations as follows.

$$\begin{cases} x_1 = y_1 \\ x_2 = y_2 \\ s_1(x_1 \oplus k_1) \oplus x_3 = y_4 \\ s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 = y_3 \\ s_1(x_1 \oplus k_1) \oplus x_3 \oplus s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 \oplus s_3(x_1 \oplus x_2 \oplus k_3) \oplus x_5 = 0 \\ s_1(x_1 \oplus k_1) \oplus x_3 \oplus s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 \oplus s_4(x_2 \oplus k_4) \oplus x_6 = 0 \\ s_1(x_1 \oplus k_1) \oplus x_3 \oplus s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 \oplus s_2(x_1 \oplus x_2 \oplus k_5) \oplus x_7 = 0 \\ s_1(x_1 \oplus k_1) \oplus x_3 \oplus s_3(x_2 \oplus k_6) \oplus x_8 = 0 \\ s_2(x_1 \oplus x_2 \oplus k_2) \oplus x_4 \oplus s_1(x_1 \oplus k_8) \oplus x_9 = 0 \end{cases}$$

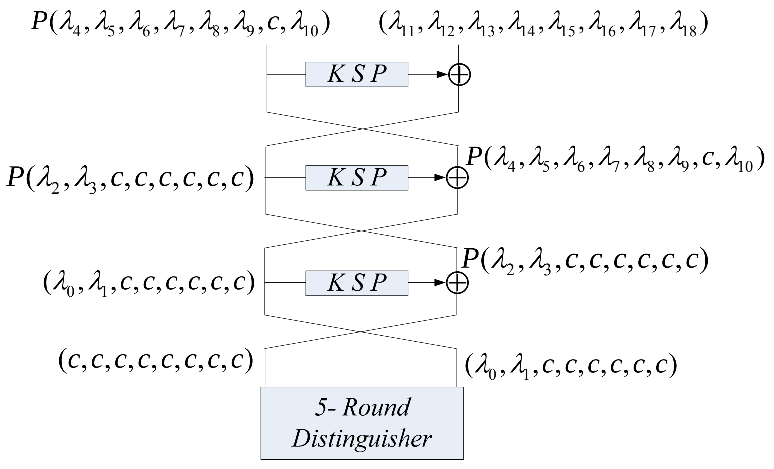
According to the simplified equations above, we find that there is only one solution. For 2^{32} values of y_3, y_4, y_1, y_2, we get a set of 2^{32} solutions, and each value of x_5, x_6, x_7, x_8, x_9 is decided by x_1, x_2, x_3, x_4. After taking over all 2^{40} values of x_5, x_6, x_7, x_8, x_9, 2^{40} sets will be obtained, i.e. 2^{72} values of $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9$ will lead to 2^{40} sets after one round encryption, in each set the bytes of y_1, y_2, y_3, y_4 are active.

3) The 2^{120} values of (X_{-2}, X_{-3}) will lead to 2^{48} sets of (X_{-1}, X_{-2}) after one round encryption, and in each set the nine bytes of $(P^{-1}(X_{-2}), P^{-1}(X_{-1}))$ are active, and other bytes are constants.
The proof of this step is just like that of the above step.

According to the above three steps, we will obtain 2^{104} sets of (X_1, X_0) from 2^{120} values of (X_{-2}, X_{-3}) described in the lemma 2. In each set the bytes of $x_{0,1}, x_{0,2}$ are active and other bytes are constant. □

In line with Lemma 1 and Lemma 2, we can construct a new 15th order 8-round distinguisher of Camellia as depicted in Theorem 1(Fig.7).

Theorem 1. Let (X_1, X_0) are input of Camellia without FL/FL^{-1}. If the bytes of $P^{-1}(X_1)_7$ are constants and other bytes of (X_1, X_0) take all values of $F_{2^8}^{15}$, then each value of $t = s_3(x_{9,6} \oplus k_{9,6}) \oplus P^{-1}(X_{10})_6$ will appear even times.



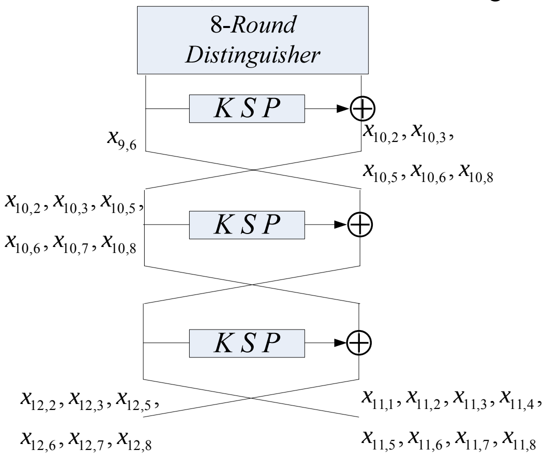
$P(\lambda_4,\lambda_5,\lambda_6,\lambda_7,\lambda_8,\lambda_9,c,\lambda_{10})$
$(\lambda_{11},\lambda_{12},\lambda_{13},\lambda_{14},\lambda_{15},\lambda_{16},\lambda_{17},\lambda_{18})$
KSP
$P(\lambda_2,\lambda_3,c,c,c,c,c,c)$
KSP
$P(\lambda_4,\lambda_5,\lambda_6,\lambda_7,\lambda_8,\lambda_9,c,\lambda_{10})$
$(\lambda_0,\lambda_1,c,c,c,c,c,c)$
KSP
$P(\lambda_2,\lambda_3,c,c,c,c,c,c)$
(c,c,c,c,c,c,c,c)
$(\lambda_0,\lambda_1,c,c,c,c,c,c)$
5- Round Distinguisher

Fig. 7. The 15th-Order 8-Round Distinguisher

4 Attacks on Reduced Round Camellia

4.1 Integral Attack on 10/11-Round Camellia-128

Based on the 15th-order 8-round distinguisher described above, we will attack 10 rounds of Camellia-128 without FL / FL^{-1} functions now, which is illustrated in Fig.8.



8-Round Distinguisher
KSP
$x_{9,6}$
$x_{10,2}, x_{10,3},$
$x_{10,5}, x_{10,6}, x_{10,8}$
$x_{10,2}, x_{10,3}, x_{10,5},$
$x_{10,6}, x_{10,7}, x_{10,8}$
KSP
KSP
$x_{12,2}, x_{12,3}, x_{12,5},$
$x_{12,6}, x_{12,7}, x_{12,8}$
$x_{11,1}, x_{11,2}, x_{11,3}, x_{11,4},$
$x_{11,5}, x_{11,6}, x_{11,7}, x_{11,8}$

Fig. 8. Integral Attacks on Reduced Round Camellia

1. Choose a structure of plaintexts (X_1, X_0). It satisfies

$$X_1 = P(\lambda_4,\lambda_5,\lambda_6,\lambda_7,\lambda_8,\lambda_9,c,\lambda_{10}),\ X_0 = P(\lambda_{11},\lambda_{12},\lambda_{13},\lambda_{14},\lambda_{15},\lambda_{16},\lambda_{17},\lambda_{18}),$$

where λ_is ($4 \le i \le 18$) are active bytes, and c is a constant. (X_1, X_0) take all values of $F_{2^8}^{15}$. Encrypt all this plaintexts and set 2^{56} counters for the seven bytes of $x_{11,6}, x_{10,2}, x_{10,3}, x_{10,5}, x_{10,6}, x_{10,7}, x_{10,8}$ and the corresponding counters plus one.

2. For the 2^{56} values of ciphertexts, there are 2^{56} values at most in bytes of $x_{11,6}, x_{10,2}, x_{10,3}, x_{10,5}, x_{10,6}, x_{10,7}, x_{10,8}$. We choose those values that the corresponding counters are odd times (the Xor value of the same value is zero).

Guessing the key bytes of $k_{10,2}, k_{10,3}, k_{10,5}, k_{10,7}, k_{10,8}$ and $k_{9,6}$, we do a partial decrypt to the single value of t. In this phase we need the partial sum technique in order to reduce the workfactor of computing the value of $s_3(x_{9,6} \oplus k_{9,6})$.

$$\begin{aligned} t &= [P^{-1}(X_{10})]_6 \oplus s_3(x_{9,6} \oplus k_{9,6}) \\ &= [P^{-1}(x_{10,2}, x_{10,3}, x_{10,5}, x_{10,6}, x_{10,8})]_6 \\ &\quad \oplus s_3[s_2(x_{10,2} \oplus k_{10,2}) \oplus s_3(x_{10,3} \oplus k_{10,3}) \oplus s_2(x_{10,5} \oplus k_{10,5}) \\ &\quad \oplus s_4(x_{10,7} \oplus k_{10,7}) \oplus s_1(x_{10,8} \oplus k_{10,8}) \oplus x_{11,6} \oplus k_{9,6}]. \end{aligned} \tag{2}$$

For the values of $x_{11,6}, x_{10,2}, x_{10,3}, x_{10,5}, x_{10,7}, x_{10,8}$, the number of which appear odd times, we do the following steps:

(a) Guess the two bytes of $k_{10,2}$ and $k_{10,3}$, and we obtain the corresponding 5 bytes value

$$(x_1, x_{10,5}, x_{10,7}, x_{10,8}, x_{11,6}).$$

(b) Guess the value of $k_{10,5}$, and compute the partial sum, then we obtain 4 bytes value

$$(x_2, x_{10,7}, x_{10,8}, x_{11,6}).$$

(c) Guess the value of $k_{10,7}$, and compute the partial sum, then we obtain 3 bytes value

$$(x_3, x_{10,8}, x_{11,6}).$$

(d) Guess the value of $k_{10,8}$, and compute the partial sum, then we obtain 2 bytes value

$$(x_4, x_{11,6}).$$

(e) Guess the value of $k_{9,6}$, and compute the partial sum, then we obtain 1 byte value

$$x_5 = s_3(x_{9,6} \oplus k_{9,6}).$$

We sum the values of x_5 and $[P^{-1}(X_{10})]_6$ over all the encryptions, and check the value of t if it is even times. If each value of t appears even times, the guessed key bytes are right, otherwise they are wrong guesses.

In Step 1, we choose 2^{120} plaintexts and need encrypt 2^{120} times. In Step 2-(a), we guess 16 bits key, and process 2^{48} ciphertexts, which cost 2^{64} S-box applications. Step 2-(b) costs workfactor as $2^{16} \times 2^{8} \times 2^{40} = 2^{64}$ at most. This is same to the other phases of Step 2, so the workfactor of Step 2 is $2^{72} \times 7$ S-box applications. There are also additional works to compute $[P^{-1}(X_{10})]_6$ in each phase of Step 2, however, using rough equivalence of 8 S-box applications to a round encryption with a new key, the work to compute $[P^{-1}(X_{10})]_6$ can be ignored. For a wrong key, the probability that it satisfies each value of t appearing even times is less than 2^{-169} (**Appendix A**). After analyzing a structure of plaintexts, there is $2^{48} \times 2^{-169} = 2^{-121}$ wrong key that will pass Step 2. So the data complexity of 2^{120} is enough and the time complexity is $2^{48} \times 2^{16} \times 5/(2^3 \times 10) = 2^{60}$ encryptions.

We also can improve this attack by adding one more round. For 11-round Camellia-128, the key schedule could be used (**Appendix B**). Guessing the key bytes of $k_{1,3}, k_{1,4}, \ldots, k_{1,8}$, and looking up S-boxes for 2^{56} values of X_1, the values of six bytes of $P^{-1}(X_0 \oplus X_2)$ will be obtained. Then choosing the six bytes of $P^{-1}(X_2)$ as constant, we can get the corresponding six bytes of $P^{-1}(X_0)$. Let the first two bytes of $P^{-1}(X_2)$ be active, and we will obtain 2^{16} values of X_0 in the chosen plaintexts. For each key of $k_{1,3}, k_{1,4}, \ldots, k_{1,8}$, encrypt all 2^{72} plaintexts. According to the key schedule, $K_{11} = K_1 <<< 60$, so the key bytes of K_{11} are known except for $k_{11,2}$, 4 bits of $k_{11,1}$ and 4 bits of $k_{11,3}$. With the partial sum technique we will decrypt three rounds to obtain the value of t. The $x_{10,i}, i = 2,3,5,6,7,8$ used in **Equ.(2)** can be represented as follows.

$$x_{10,2} = s_1(x_{11,1} \oplus k_{11,1}) \oplus s_2(x_{11,2} \oplus k_{11,2}) \oplus c_2$$

$$x_{10,3} = s_1(x_{11,1} \oplus k_{11,1}) \oplus s_2(x_{11,2} \oplus k_{11,2}) \oplus s_3(x_{11,3} \oplus k_{11,3}) \oplus c_3$$

$$x_{10,5} = s_1(x_{11,1} \oplus k_{11,1}) \oplus s_2(x_{11,2} \oplus k_{11,2}) \oplus c_5$$
$$x_{10,6} = s_2(x_{11,2} \oplus k_{11,2}) \oplus s_3(x_{11,3} \oplus k_{11,3}) \oplus c_6$$
$$x_{10,7} = s_3(x_{11,3} \oplus k_{11,3}) \oplus c_7$$
$$x_{10,8} = s_1(x_{11,1} \oplus k_{11,1}) \oplus c_8$$

where $c_i, i = 2,3,5,6,7,8$ should be pre-computed in order to reduce the time complexity. For example, $c_2 = s_4(x_{11,4} \oplus k_{11,4}) \oplus s_2(x_{11,5} \oplus k_{11,5}) \oplus s_4(x_{11,7} \oplus k_{11,7}) \oplus s_1(x_{11,8} \oplus k_{11,8}) \oplus x_{12,2}$. The time complexity of pre-computation is about $2^{48} \times 2^{72} \times \frac{2}{3} \approx 2^{119.4}$ rounds encryption, which can be ignored compared with the consuming later. For the 2^{72} ciphertexts, we do the following steps.

1. Guess $k_{11,2}$ and 4 bits of $k_{11,3}$, and we obtain the corresponding 8 bytes value

$$(x_{11,1}, x_{11,6}, c'_2, c'_3, c'_5, x_{10,6}, x_{10,7}, c_8)$$

where $c'_2 = s_2(x_{11,2} \oplus k_{11,2}) \oplus c_2$, and c'_3, c'_5 are similar.

2. Guess 4 bits of $k_{11,1}$,and we obtain the corresponding 7 bytes value

$$(x_{10,2}, x_{10,3}, x_{10,5}, x_{10,6}, x_{10,7}, x_{10,8}, x_{11,6})$$

The remainder two rounds decryptions are just as the above attack on 10-round Camellia. The computation required in Step 1 is the main workfactor, which is equivalent to $2^{48} \times 2^{72} \times 2^{12}$ $= 2^{132}$ S-box lookups. The total time complexity is $2^{132} / (2^3 \times 3) \approx 2^{125.5}$ encryptions.

4.2 Integral Attack on Camellia-192/256

In this subsection, we describe improved integral attacks on 11-round Camellia-192 and 12-round Camellia-256. The key schedule can't be used here. We add 1 round after 10-round Camellia, and guess all bytes of K_{11}, and decrypt the last round, where the workfactor is the equivalent of $2^{64} \times 2^{64} = 2^{128}$ F function operations and $2^{64} \times 2^{48} \times 2^{48} = 2^{160}$ Xor operations. The rest workfactor can be ignored, so the main time complexity of the attack on 11 rounds Camellia-192/256 is $2^{160} / (2^6 \times 11) \approx 2^{150.5}$. In a similar way the time complexity of attack on 12 rounds Camellia-256 is $2^{214.3}$. There no relation of subkeys can be used.

5 Conclusion

The improved integral attacks on reduced round Camellia were described in this paper. We gave a method of forward-extending the length of integral distinguisher, by which a new 8-round integral distinguisher for Camellia was proposed. And then we attacked 11-round Camellia-128 and 11/12-round Camellia-192/256 with the partial sum technique. Table 2 summarizes our integral attacks together with the previously known integral-like attacks on Camellia.

Table 2. Results of Integral-like Attacks on Camellia

Camellia-*b*	Rounds	FL/FL^{-1}	Method	D-Rounds	Data	Time	Notes
Camellia-128	8	Yes	SA	4	2^{48}	2^{116}	[19]
	9	No	SLA	5	2^{66}	$2^{84.8}$	[3]
	10	No	IntA	8	2^{120}	2^{120}	Sec4.1
	11	No	IntA	8	2^{120}	$2^{123.9}$	Sec4.1
Camellia-192 /256	9	No	HODC	5	2^{21}	2^{188}	[5]
	9	Yes	SA	4	$2^{60.5}$	2^{202}	[19]
	10	No	SLA	5	2^{66}	$2^{167.3}$	[3]
	11	No	SLA	5	2^{66}	$2^{211.6}$	[3]
	11	N/Y	HODC	5	2^{66}	$2^{255.6}$	[5]
	11	No	IntA	8	2^{120}	$2^{150.5}$	Sec4.2
	12	No	SLA	5	2^{66}	$2^{249.6}$	[3]
	12	No	IntA	8	2^{120}	$2^{214.3}$	Sec4.2

Note 1. D-Rounds: Distinguisher Rounds; SA: Square Attack; IntA: Integral Attack；
HODC: Higher Order Differential Attack; SLA: Square Like Attack.
Note 2. Time complexity is measured in encryption units.

According to Table 2, the integral attacks presented in this paper make significant improvements on both data and time complexities. However, the full round Camellia provides a sufficient safety margin. Our new method also can be used for any Feistel-SP structure. How to evaluate the security of Feistel-SP structure against integral attack is more important, which will be our future work.

References

1. Aoki, K., Ichikawa, T., Kanda, M., et al.: *Camellia: A 128-Bit block cipher suitable for multiple platforms design and analysis*. In: Ito, T., Abadi, M. (eds.) TACS 1997. LNCS, vol. 1281, pp. 39-56. Springer, Heidelberg (1997)
2. Daemen, J., Knudsen, L., and Rijmen, V.: *The block cipher Square*. In: Biham, E.(ed.) FSE 1997. LNCS, vol. 1267, pp. 149-165. Springer, Heidelberg (1997)
3. Duo, L., Li, C., and Feng, K.Q.: *Square Like Attack on Camellia*. In: Qing, S., Imai, H., and Wang, G.(eds.) ICICS 2007. LNCS, vol. 4861, pp. 269-283. Springer, Heidelberg (2007)
4. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: *Improved cryptanalysis of Rijndael*. In: Schneier, B.(ed.) FSE 2000. LNCS, vol. 1978, pp. 213-230. Springer, Heidelberg (2001)
5. Hatano, Y., Sekine, H., Kaneko, T.: *Higher order differential attack of Camellia (II)*. In: Nyberg, K., Heys, H.M.(eds.) SAC 2002. LNCS, vol. 2595, pp. 129-164. Springer, Heidelberg (2003)
6. International Standardization of Organization (ISO), International Standard-ISO/IEC 18033-3, Information technology-Security techniques-Encryption algorithms-Part 3: Block ciphers, July, 2005.
7. Kanda, M., Matsumoto., T.: *Security of Camellia against truncated differential cryptanalysis*. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 119-137. Springer, Heidelberg (2002)
8. Knudsen, L. and Wagner, D.: *Integral cryptanalysis*. In: Daemen, J., Rijmen, V.(eds.) FSE 2002. LNCS, vol. 2365, pp. 112-127. Springer, Heidelberg (2002)
9. Lee, S., Hong, S., Lee, S., Lim, J., and Yoon, S.: *Truncated differential cryptanalysis of Camellia*. In: Kim, K.-c.(ed.) ICISC 2001. LNCS, vol. 2288, pp. 32-38. Springer, Heidelberg (2002)
10. Li, Y., Wu, W., and Zhang, L.: *Integral Attacks on Reduced-Round ARIA Block Cipher*. In: Kwak, J., et al. (eds.) ISPEC 2010. LNCS, vol. 6047, pp. 19-29, Springer, Heidelberg (2010)
11. Li, Y., Wu, W., Zhang, L., and Zhang, L.: *Improved Integral Attack on Rijndael*. Journal of Information Science and Engineering (JISE), to appear.
12. Reza Z'aba, M., and Raddum, H., Henricksen, M., and Dawson, E.: *Bit-Pattern Based Integral Attack*. In: Nyberg, K. (ed.), FSE 2008. LNCS, vol. 5086, pp. 363-381. Springer, Heidelberg (2008)
13. Shirai, T., Kanamaru, S., and Abe, G.: *Improved upper bounds of differential and linear characteristic probability for Camellia*. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 147-151. Springer, Heidelberg (2002)
14. Shirai, T.: *Differential, linear, boomerang and rectangle cryptanalysis of reduced-round Camellia*. In: Proceedings of 3rd NESSIE workshop (November 2002)
15. Sugita, M., Kobara, K., and Imai, H.: *Security of reduced version of the block cipher Camellia against truncated and impossible differential cryptanalysis*. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 193-207. Springer, Heidelberg (2001)
16. Wu, W., Feng, D., and Chen, H.: *Collision attack and pseudorandomness of reduced round Camellia*. In: Handschuh, H., Hasan, M.A.(eds.) SAC 2004. LNCS, vol. 3357, pp. 256-270. Springer, Heidelberg (2004)
17. Wu, W., Zhang, W. and Feng, D.: *Impossible differential cryptanalysis of Reduced-Round ARIA and Camellia*. In: Journal of Compute Science and Technology 22(3), pp. 449-456, Springer, Heidelberg (2007)
18. Wu, W., Zhang, L., and Zhang, W.: *Improved impossible differential cryptanalysis of reduced-round Camellia*. In: Avanzi, R., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 442-456. Springer, Heidelberg(2009)
19. Yeom, Y., Park, S., and Kim, I.: *On the security of Camellia against the Square attack*. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 89-99. Springer, Heidelberg (2002)
20. Yeom, Y., Park, S., and Kim, I.: *A study of integral type cryptanalysis on Camellia*. In Proceedings of the 2003 Symposium on Cryptography and Information Security, pp. 453-456 (2003)

A *The probability of* 2^{-169}

The probability is obtained from the following analysis. For 2^{56} times appearing of *t*, we need study the number of each different value appearing even times according to the following situations. First, if each value appears 2 times, the number of arrangements is

$$\binom{256}{128} \times \left(\frac{256!}{2!2!\cdots 2!}\right) \times \left(\frac{128!}{128!}\right).$$

Second, if one value appears 4 times and each other value appears 2 times, the number of arrangements is

$$\binom{256}{127} \times \left(\frac{256!}{4!2!\cdots 2!}\right) \times \left(\frac{127!}{1!126!}\right).$$

Likewise, if there are *i* different values appearing, the number of each different value appearing even times is

$$\binom{256}{i} \times \left(\frac{256!}{(2j_1)!(2j_2)!\cdots(2j_i)!}\right) \times \left(\frac{i!}{n_1!n_2!\cdots n_l!}\right),$$

where $(j_1, j_2, \cdots, j_i)$ is the partition of *i* parts of 128, and n_1 denotes the appearing times of j_1 in the set of $(j_1, j_2, \cdots, j_i)$. The vector $(n_0, n_1, \ldots, n_l)$ satisfies the equation $n_1 + n_2 + \ldots + n_l = i$, i.e. there are l different j_ms ($m = 1, 2, \ldots, i$) in $(j_1, j_2, \cdots, j_i)$. When *i* takes all values from 128 to 1, we will obtain all the number of arrangements, the sum of which is defined as

$$f(128) = \bigoplus_{i=1}^{128} \binom{256}{i} \times \left(\frac{256!}{(2j_1)!(2j_2)!\cdots(2j_i)!}\right) \times \left(\frac{i!}{n_1!n_2!\cdots n_l!}\right).$$

By searching with computer, we only can obtain

$$f(64) = \bigoplus_{i=1}^{64} \binom{256}{i} \times \left(\frac{128!}{(2j_1)!(2j_2)!\cdots(2j_i)!}\right) \times \left(\frac{i!}{n_1!n_2!\cdots n_l!}\right) \approx 2^{855}.$$

However, it is easy to get the inequation $f(128) < f(64) \times 256^{128}$. Therefore, for 256 times appearing of *t*, the probability of each different value appearing even times is less than $2^{855} \times 256^{128} / 256^{256} = 2^{-169}$.

B *The Relation of Subkeys*

The relationship of subkeys for Camellia-128 is listed in Table 3.

Table 3. Subkeys for 128-bit keys

128-bit keys	subkey	value
Prewhitening	kw_1	$(K_L{<}{<}{<}0)_L$
	kw_2	$(K_L{<}{<}{<}0)_R$
F(Round 1)	k_1	$(K_A{<}{<}{<}0)_L$
F(Round 2)	k_2	$(K_A{<}{<}{<}0)_R$
F(Round 3)	k_3	$(K_L{<}{<}{<}15)_L$
F(Round 4)	k_4	$(K_L{<}{<}{<}15)_R$
F(Round 5)	k_5	$(K_A{<}{<}{<}15)_L$
F(Round 6)	k_6	$(K_A{<}{<}{<}15)_R$
FL	kl_1	$(K_A{<}{<}{<}30)_L$
FL^{-1}	kl_2	$(K_A{<}{<}{<}30)_R$
F(Round 7)	k_7	$(K_L{<}{<}{<}45)_L$
F(Round 8)	k_8	$(K_L{<}{<}{<}45)_R$
F(Round 9)	k_9	$(K_A{<}{<}{<}45)_L$
F(Round 10)	k_{10}	$(K_L{<}{<}{<}60)_R$
F(Round 11)	k_{11}	$(K_A{<}{<}{<}60)_L$
F(Round 12)	k_{12}	$(K_A{<}{<}{<}60)_R$
FL	kl_3	$(K_L{<}{<}{<}77)_L$
FL^{-1}	kl_4	$(K_L{<}{<}{<}77)_R$
F(Round 13)	k_{13}	$(K_L{<}{<}{<}94)_L$
F(Round 14)	k_{14}	$(K_L{<}{<}{<}94)_R$
F(Round 15)	k_{15}	$(K_A{<}{<}{<}94)_L$
F(Round 16)	k_{16}	$(K_A{<}{<}{<}94)_R$
F(Round 17)	k_{17}	$(K_L{<}{<}{<}111)_L$
F(Round 18)	k_{18}	$(K_L{<}{<}{<}111)_R$
Postwhitening	kw_3	$(K_A{<}{<}{<}111)_L$
	kw_4	$(K_A{<}{<}{<}111)_R$