New Results on Impossible Differential Cryptanalysis of Reduced–Round Camellia–128

Hamid Mala[1], Mohsen Shakiba[1], Mohammad Dakhilalian[1], and Ghadamali Bagherikaram[2]

[1] Cryptography & System Security Research Laboratory, Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, Iran
{hamid_mala@ec, m.shakiba@ec, mdalian@cc}.iut.ac.ir
[2] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada
gbagheri@cst.uwaterloo.ca

Abstract. Camellia, a 128–bit block cipher which has been accepted by ISO/IEC as an international standard, is increasingly being used in many cryptographic applications. In this paper, using the redundancy in the key schedule and accelerating the filtration of wrong pairs, we present a new impossible differential attack to reduced–round Camellia. By this attack 12–round Camellia–128 without FL/FL^{-1} functions and whitening is breakable with a total complexity of about $2^{116.6}$ encryptions and $2^{116.3}$ chosen plaintexts. In terms of the numbers of the attacked rounds, our attack is better than any previously known attack on Camellia–128.

1 Introduction

Camellia [1] is a 128–bit block cipher that supports several key lengths. For the sake of simplicity, Camellia with n–bit keys is denoted by Camellia–n, n=128, 192, 256. Camellia was jointly proposed in 2000 by NTT and Mitsubishi and then was submitted to several standardization and evaluation projects. It was selected as a winner of CRYPTREC e-government recommended ciphers in 2002 [5], NESSIE block cipher portfolio in 2003 [17] as well as the standardization activities at IETF [18]. Finally Camellia was selected as an international standard by ISO/IEC in 2005 [9]. As one of the most widely used block ciphers, Camellia has received a significant amount of cryptanalytic attention. The most efficient cryptanalytic results on Camellia include linear and differential attacks [19], truncated differential attack [5,10,13,20], higher order differential attack [7,11], collision attack [14,21], square attack [8,14,24], a square like attack [6] and impossible differential attack [15,20,22,23].

Impossible differential cryptanalysis, an extension of the differential attack [4], is one of the most powerful methods used for block cipher cryptanalysis. This method was first introduced by Biham [3] and Knudsen [12] independently. Impossible differential attacks use differentials that hold with probability zero (impossible differentials) to eliminate the wrong keys and leave the right key.

M.J. Jacobson Jr., V. Rijmen, and R. Safavi-Naini (Eds.): SAC 2009, LNCS 5867, pp. 281–294, 2009.

The most efficient impossible differential attacks, recently proposed to reduced variants of Camellia, are as follows. The initial analysis of the security of Camellia to impossible differential cryptanalysis was given in [20]. They presented some 7–round impossible differentials for Camellia. In [23] Wu et al. introduced a nontrivial 8–round impossible differential that lead to an impossible differential attack on Camellia–192 and Camellia–256 without the FL/FL^{-1} functions with complexity of about 2^{118} chosen plaintexts and a time complexity of about 2^{126} memory accesses. Introducing the early abort technique, Lu et al. improved the impossible differential attack on Camellia in [16]. Later in [22] Wu et al. found a flaw in [16] and presented an impossible differential attack on 12–round Camellia–128 and claimed that their attack has a data complexity of 2^{65} chosen plaintexts and a time complexity of about $2^{111.5}$ encryptions. In this paper, we point out a flaw in their attack and show that its time complexity is more than exhaustive key search. However, their work is the first impossible differential attack on Camellia that considers the weakness in its key schedule.

In this paper, using the same 8–round impossible differential of [23], considering the weakness in the key schedule of Camellia–128, and also exploiting a hash table to simplify the selection of proper pairs, we present the first successful 12–round attack on Camellia–128. The proposed attack requires $2^{116.3}$ chosen plaintexts and has a total time complexity equivalent to about $2^{116.6}$ encryptions. We summarize our results along with previously known results on Camellia–128 in Table 1. The results of [16] in Table 1 come from its early version reported in Lu's PhD thesis [15], so we mark them with "†". In this table, time complexity is measured in encryption units unless MA is mentioned for memory accesses.

The rest of this paper is organized as follows: Section 2 provides a brief description of Camellia. We propose our new impossible differential attack on 12–round Camellia–128 in Section 3. Section 3 includes the previously known 8–round impossible differential (in Subsection 3.1), some observations on the key schedule of Camellia–128 (in Subsection 3.2), the proposed attack procedure on 12–round

Table 1. Summary of previous attacks and our new attack on Camellia–128

#Rounds	FL/FL^{-1}	Data	Time	Attack type	Source
8	no	$2^{83.6}$	$2^{55.6}$	Truncated Diff.	[13]
8	no	2^{20}	2^{120}	Higher Order Diff.	[7]
9	no	2^{92}	2^{111}	Higher Order Diff.	[7]
9	yes	2^{48}	2^{122}	Square.	[14]
9	no	$2^{113.6}$	2^{121}	Collision.	[21]
9	no	2^{88}	2^{90}	Square.	[14]
9	no	2^{105}	2^{105}	Differential.	[19]
9	no	2^{66}	$2^{84.8}$	Square like.	[6]
10	no	2^{120}	2^{121}	Linear.	[19]
11	no	2^{118}	2^{126}MA&2^{118}	Impossible Diff.	[16]†
11	no	2^{118}	2^{126}MA	Impossible Diff.	[16]†
12	no	$2^{116.3}$	$2^{116.6}$	Impossible Diff.	This work

Camellia–128 (in Subsection 3.3), and the analysis of the attack complexity (in Subsection 3.4). Finally, we conclude the paper in Section 4.

2 Preliminaries

2.1 Notations

In this paper, we will use the following notations:

L^{r-1}	: the left 64–bit half of the r–th round input,
R^{r-1}	: the right 64–bit half of the r–th round input,
k^r	: the subkey used in r–th round,
k_l^r	: the l–th byte of a subkey k^r,
$k_l^r[i-j]$	: the i–th to the j–th bits of $k_l^r, i, j = 1, 2, ..., 8, i \leq j$,
$x\|y$	: bit string concatenation of x and y,
$\oplus$	: bit-wise exclusive or operation,
$x <<<_l$	: the rotation of x by l bits to the left.

2.2 Description of Camellia

The 128–bit block cipher Camellia [1] has an 18–round (for 128 bit keys) or 24–round (for 192/256–bit keys) Feistel structure. The FL/FL^{-1} functions layer is inserted every 6 rounds. Before the first round and after the last round, there are pre– and post–whitening layers. In this paper we will consider a reduced variant of Camellia without FL/FL^{-1} functions and whitening layers. The Feistel structure of the r–th round is

$$L^r = R^{r-1} \oplus F(L^{r-1}, k^r)\ ,\ R^r = L^{r-1},$$

where function F consists of a key–addition layer, a substitution transformation S and a diffusion layer P. The S transformation contains 4 types of 8×8 S–boxes s_1, s_2, s_3 and s_4 as follows:

$$\begin{aligned} &S(x_1|x_2|x_3|x_4|x_5|x_6|x_7|x_8) \\ &= s_1(x_1)|s_2(x_2)|s_3(x_3)|s_4(x_4)|s_2(x_5)|s_3(x_6)|s_4(x_7)|s_1(x_8). \end{aligned}$$

The transformation $P : (\{0,1\}^8)^8 \rightarrow (\{0,1\}^8)^8$ maps $(z_1, ..., z_8)$ to $(z'_1, ..., z'_8)$. This transformation and its inverse, P^{-1}, are defined as:

$$\begin{array}{ll} z'_1 = z_1 \oplus z_3 \oplus z_4 \oplus z_6 \oplus z_7 \oplus z_8 & z_1 = z'_2 \oplus z'_3 \oplus z'_4 \oplus z'_6 \oplus z'_7 \oplus z'_8 \\ z'_2 = z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8 & z_2 = z'_1 \oplus z'_3 \oplus z'_4 \oplus z'_5 \oplus z'_7 \oplus z'_8 \\ z'_3 = z_1 \oplus z_2 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_8 & z_3 = z'_1 \oplus z'_2 \oplus z'_4 \oplus z'_5 \oplus z'_6 \oplus z'_8 \\ z'_4 = z_2 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7 & z_4 = z'_1 \oplus z'_2 \oplus z'_3 \oplus z'_5 \oplus z'_6 \oplus z'_8 \\ z'_5 = z_1 \oplus z_2 \oplus z_6 \oplus z_7 \oplus z_8 & z_5 = z'_1 \oplus z'_2 \oplus z'_5 \oplus z'_7 \oplus z'_8 \\ z'_6 = z_2 \oplus z_3 \oplus z_5 \oplus z_7 \oplus z_8 & z_6 = z'_2 \oplus z'_3 \oplus z'_5 \oplus z'_6 \oplus z'_8 \\ z'_7 = z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_8 & z_7 = z'_3 \oplus z'_4 \oplus z'_5 \oplus z'_6 \oplus z'_7 \\ z'_8 = z_1 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7 & z_8 = z'_1 \oplus z'_4 \oplus z'_6 \oplus z'_7 \oplus z'_8 \end{array}$$

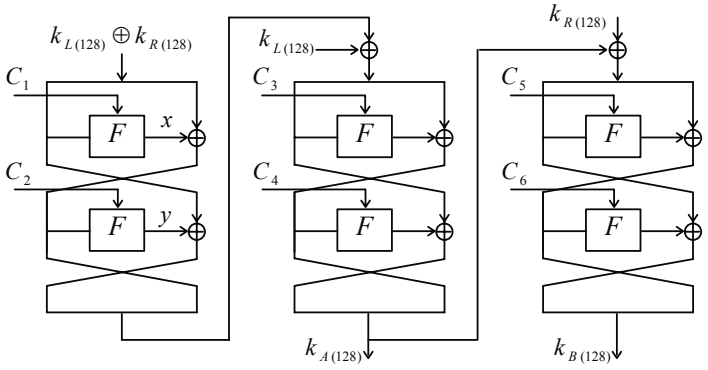

Fig. 1. Key schedule of Camellia

Table 2. The first 12 round keys of Camellia–128

Round	Subkey	Value	Round	Subkey	Value
1	k^1	$(k_A <<<_0)_L$	7	k^7	$(k_L <<<_{45})_L$
2	k^2	$(k_A <<<_0)_R$	8	k^8	$(k_L <<<_{45})_R$
3	k^3	$(k_L <<<_{15})_L$	9	k^9	$(k_A <<<_{45})_L$
4	k^4	$(k_L <<<_{15})_R$	10	k^{10}	$(k_L <<<_{60})_R$
5	k^5	$(k_A <<<_{15})_L$	11	k^{11}	$(k_A <<<_{60})_L$
6	k^6	$(k_A <<<_{15})_R$	12	k^{12}	$(k_A <<<_{60})_R$

Fig. 1 shows the key schedule of Camellia. For Camellia–128, two 128–bit variables k_L and k_R are defined as follows. The 128–bit user key is used as k_L, and k_R is a 128–bit string of 0 bits. Two 128–bit variables k_A and k_B are generated from k_L and k_R as shown in Fig. 1, in which $C_i, i = 1, ..., 6$ are constants used as the keys of the Feistel round function. The round keys of Camellia are rotations of variables k_A, k_B, k_L and k_R. Note that k_B is used only if the length of the user key is 192 or 256 bits. Here, we only give the first 12 round keys for Camellia–128 in Table 2.

2.3 Analysis of Wu et al.'s Attack on Camellia–128

In step (3.c.iii) of Section 4.1 in [22], the authors write: "Furthermore, the probability that a subkey guess may remain after this test is about $(1 - 2^{-8})$." At the first look, it seems to be true, but we show that this statement and thus the resulted complexity are not true. We show that the correct value for this probability is $(1 - 2^{-68})$, and also we calculate the dominant part of the time complexity of the attack on Camellia–128 proposed in [22].

At the end of step (3.b) there remain 2^{5+m} pairs. Below, we specify the list and the number of subkeys that are determined for each of these pairs:

1. Only one value for subkey bytes $(k_1^1, k_2^1, k_3^1, k_5^1, k_8^1)$ and $(k_1^{12}, k_2^{12}, k_3^{12}, k_5^{12}, k_8^{12})$ that satisfy the required differences in Round 1 and 12, and the 28–bit condition suggested by Property 1–1,
2. 2^{16} guesses of the 16 unknown bits $(k_4^1[1-4], k_6^1, k_7^1[1-4])$,
3. according to Property 1-3, only one value for the $(k_4^{12}, k_6^{12}, k_7^{12})$,
4. only one value for k_1^2 which is obtained from the difference distribution table of S-boxes, and
5. 2^{-8} value for k_1^{11}, because the only value obtained for $(k_1^{11}$ in step (c.ii) must also satisfy the 8–bit condition $k_1^{11} = (k_8^1[5-8]|k_1^2[1-4])$.

Thus, the number of 76–bit target subkeys that satisfy the impossible differential for each of the 2^{5+m} remaining pairs is $1 \times 1 \times 2^{16} \times 1 \times 1 \times 2^{-8} = 2^8$. Thus, the probability that a 76–bit target subkey guess be discarded by each of these 2^{5+m} pairs is $\frac{2^8}{2^{76}} = 2^{-68}$. Hence the number of 76–bit wrong subkeys remained at the end of the attack procedure is $(2^{76}-1).(1-2^{-68})^{2^{m+5}}$. If we choose $m = 9$, as [22] proposes, the number of remaining wrong subkeys becomes:

$$(2^{76}-1).(1-2^{-68})^{2^{14}} \approx 2^{76}.e^{-2^{-54}} \approx 2^{76}$$

If we accept that only one wrong subkey remains, m can be obtained as below:

$$(2^{76}-1).(1-2^{-68})^{2^{m+5}} = 1 \Rightarrow m \approx -5 + 68 + \log_2(\frac{76}{\log_2 e}) \approx 68.7$$

Thus, the number of the required chosen plaintexts is about $2^{m+56} = 2^{124.7}$. Also the dominant part of time complexity which is related to step 2, will be about $2^{50} \times 2^{124.7} = 2^{174.7}$ memory accesses. Hence, this attack is infeasible. It seems that there is a similar mistake in computing the complexity of the attack on Camellia-256 proposed in [22].

3 Impossible Differential Cryptanalysis of Reduced Camellia–128

In this section, we first present the 8–round impossible differential of Camellia introduced in [23], then we propose an impossible differential attack on 12–round Camellia–128 without the FL/FL^{-1} functions. Finally, we analyze the complexity of our attack in Section 3.4.

3.1 8–Round Impossible Differentials of Camellia

In 2007, Wu et al. [23] found the following 8–round impossible differentials of Camellia: $(0|0|0|0|0|0|0|0, a|0|0|0|0|0|0|0) \rightarrow_8 (h|0|0|0|0|0|0|0, 0|0|0|0|0|0|0|0)$, where a and h are any two non–zero bytes. Fig. 2 illustrates more details. A detailed explanation of these 8–round impossible differentials is given in [23].

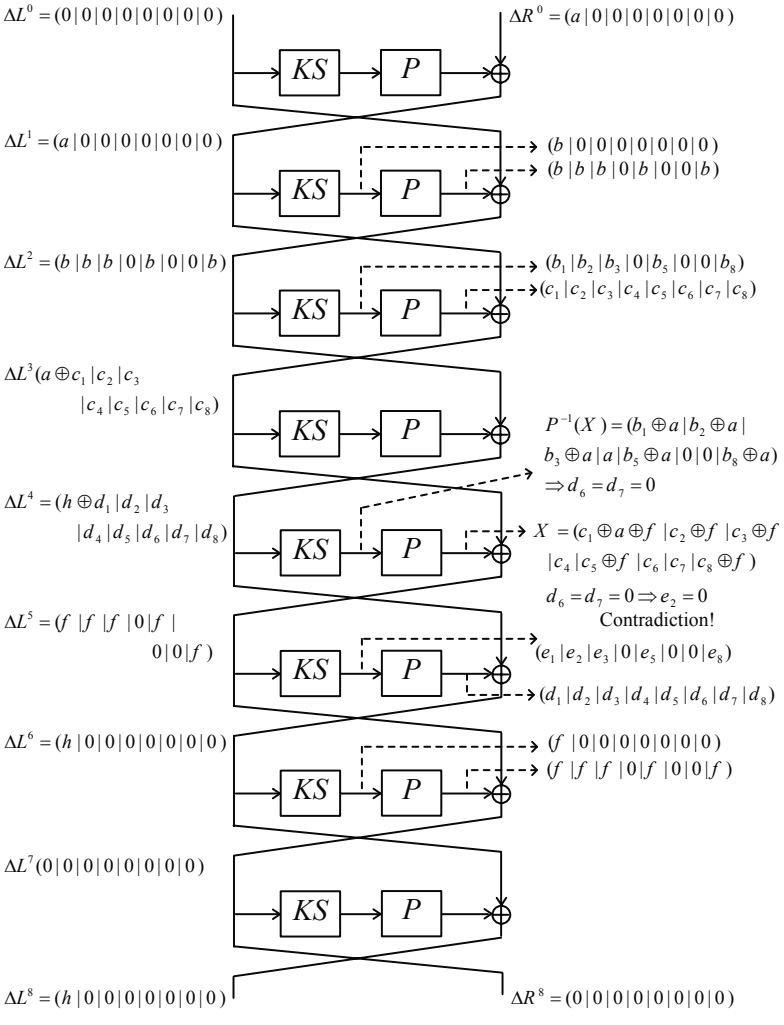

Fig. 2. 8–round impossible differentials of Camellia

3.2 Some Observations on the Key Schedule of Camellia–128

Redundancy in the Key Schedule: We first consider the relation between the target subkeys in our attack. The 18–byte target subkeys include the 8 bytes of k^1, the byte k_1^2, the byte k_1^{11} and 8 bytes of the last round key, k^{12}. Considering the key schedule of Camellia–128, we immediately observe that these 144 target bits are not distinct. From Table 2 we know that the four additional round keys k^1, k^2, k^{11}, k^{12} are rotations of the intermediate value k_A, as below:

$$k^1 = (k_A <<<_0)_L \quad , \quad k^2 = (k_A <<<_0)_R,$$
$$k^{11} = (k_A <<<_{60})_L \text{ and } k^{12} = (k_A <<<_{60})_R$$

Table 3. Target subkeys in the attack represented in Fig. 4

Target Byte	Equivalent 8 bits of k_A	Target Byte	Equivalent 8 bits of k_A
k_1^1	$k_1\|k_2...\|k_8$	k_1^{11}	$k_{61}\|k_{62}...\|k_{68}$
k_2^1	$k_9\|k_{10}...\|k_{16}$	k_1^{12}	$k_{125}\|...\|k_{128}\|k_1\|...\|k_4$
k_3^1	$k_{17}\|k_{18}...\|k_{24}$	k_2^{12}	$k_5\|k_6...\|k_{12}$
k_4^1	$k_{25}\|k_{26}...\|k_{32}$	k_3^{12}	$k_{13}\|k_{14}...\|k_{20}$
k_5^1	$k_{33}\|k_{34}...\|k_{40}$	k_4^{12}	$k_{21}\|k_{22}...\|k_{28}$
k_6^1	$k_{41}\|k_{42}...\|k_{48}$	k_5^{12}	$k_{29}\|k_{30}...\|k_{36}$
k_7^1	$k_{49}\|k_{50}...\|k_{56}$	k_6^{12}	$k_{37}\|k_{38}...\|k_{44}$
k_8^1	$k_{57}\|k_{58}...\|k_{64}$	k_7^{12}	$k_{45}\|k_{46}...\|k_{52}$
k_1^2	$k_{65}\|k_{66}...\|k_{72}$	k_8^{12}	$k_{53}\|k_{54}...\|k_{60}$

Let us denote the intermediate value k_A by its bits as $k_A = k_1|k_2|...|k_{128}$. Then we can distinguish 18 target subkey bytes in bit strings of k_A in Table 3. It is obvious that the 18 target bytes are composed of only 76 distinct bits. This fact will help us to reduce the complexity of our attack. These distinct 76 bits include $k_1|k_2|...|k_{72}$ in Rounds 1, 2 and the four bits $k_{125}|...|k_{128}$ in the last round. This fact has previously been considered in [22].

Relation between k_L and k_A: Since in our attack some bits of k_A are recovered, here we investigate the relation between the master key of Camellia–128, $k_L = k_L^L|k_L^R$ and the intermediate key value $k_A = k_A^L|k_A^R$. In other words, we will show that k_L can be extracted from k_A. According to the key schedule of Camellia–128, k_R is zero. Let the outputs of round functions F in first and second rounds of the key schedule be denoted by x and y, respectively. According to Fig. 1, we can obtain x and y as functions of only k_A as below:

$$\begin{aligned} y &= (k_A^L \oplus F_{C_4}(k_A^R) \oplus k_L^L) \oplus k_L^L = k_A^L \oplus F_{C_4}(k_A^R) \\ x &= (k_A^R \oplus F_{C_3}(k_A^L \oplus F_{C_4}(k_A^R)) \oplus k_L^R) \oplus k_L^R = k_A^R \oplus F_{C_3}(k_A^L \oplus F_{C_4}(k_A^R)) \\ &= k_A^R \oplus F_{C_3}(y) \end{aligned}$$

In a same way k_L can be represented in terms of x and y as below:

$$k_L^R = F_{C_2}^{-1}(y) \oplus x \quad k_L^L = F_{C_1}^{-1}(x)$$

So according to above equations we can obtain the master key of Camellia–128, $k_L = k_L^L|k_L^R$ in terms of k_A as below:

$$\begin{aligned} k_L^R &= F_{C_2}^{-1}(k_A^L \oplus F_{C_4}(k_A^R)) \oplus k_A^R \oplus F_{C_3}(k_A^L \oplus F_{C_4}(k_A^R)) \\ k_L^L &= F_{C_1}^{-1}(k_A^R \oplus F_{C_3}(k_A^L \oplus F_{C_4}(k_A^R))) \end{aligned}$$

Hence, the complexity of obtaining k_L from k_A is about four 1–round Camellia encryptions.

3.3 Impossible Differential Attack on 12–Round Camellia–128

In this section, we present the first successful impossible differential attack on 12 rounds of Camellia–128 without the FL/FL^{-1} functions and whitening. We

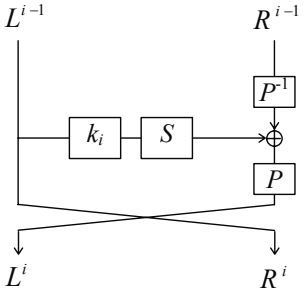

Fig. 3. Equivalent structure for one round of Camellia

attack Rounds 1 to 12, and use the 8–round impossible differential in Rounds 3 to 10. The attack is illustrated in Fig. 4. For the sake of simplicity, in Fig. 4 we use the equivalent round functions of Camellia in the Rounds 1, 2 and 12. The equivalent round function, which is shown in Fig. 3, is obtained by moving the P function to the output of the XOR operation and applying a transformation P^{-1} to the data line entering the XOR operation. According to Fig. 3, the equivalence of this modified structure to the original version can be verified easily as below:

$$\begin{aligned} R^i &= L^{i-1}, \\ L^i &= P(S(k^i \oplus L^{i-1}) \oplus P^{-1}(R^{i-1})) \\ &= P(S(k^i \oplus L^{i-1})) \oplus R^{i-1} \\ &= F(L^{i-1}, k^i) \oplus R^{i-1} \end{aligned}$$

In a traditional impossible differential attack where there exist additional rounds on both sides of the impossible differential, the attacker first checks a series of conditions in one side and choose pairs (or keys) that satisfy these conditions. She moves to the other side when she finishes checking all the conditions in the first side. When analyzing the Camellia, we observed that its structure allows us to change the side before finishing the investigation of all the conditions of one side. Thus we can check the condition that filters a greater number of pairs (or keys) before the other conditions. This strategy reduces the time complexity without any effect on the data complexity. So in the proposed attack, we first check some conditions in Round 1, then we conduct the attack in Rounds 12 and 11, and then we return to Rounds 1 and 2.

The attack procedure is as follows:

1. Take 2^n structures of plaintexts such that each structure contains 2^{56} plaintexts $P_i = L_i^0 | R_i^0$ with:

$$\begin{aligned} L_i^0 &= (a'|a'|a'|\alpha_4|a'|\alpha_6|\alpha_7|a'), \\ R_i^0 &= P(y_1'|y_2'|y_3'|\beta_4|y_5'|\beta_6|\beta_7|y_8') \oplus (y'|\gamma_2|\gamma_3|\gamma_4|\gamma_5|\gamma_6|\gamma_7|\gamma_8) \end{aligned}$$

   where the 7 bytes $(a', y', y_1', y_2', y_3', y_5', y_8')$ take all the possible values, and the bytes with the forms $\alpha_\times, \beta_\times$ and $\gamma_\times$ are fixed values in each structure.

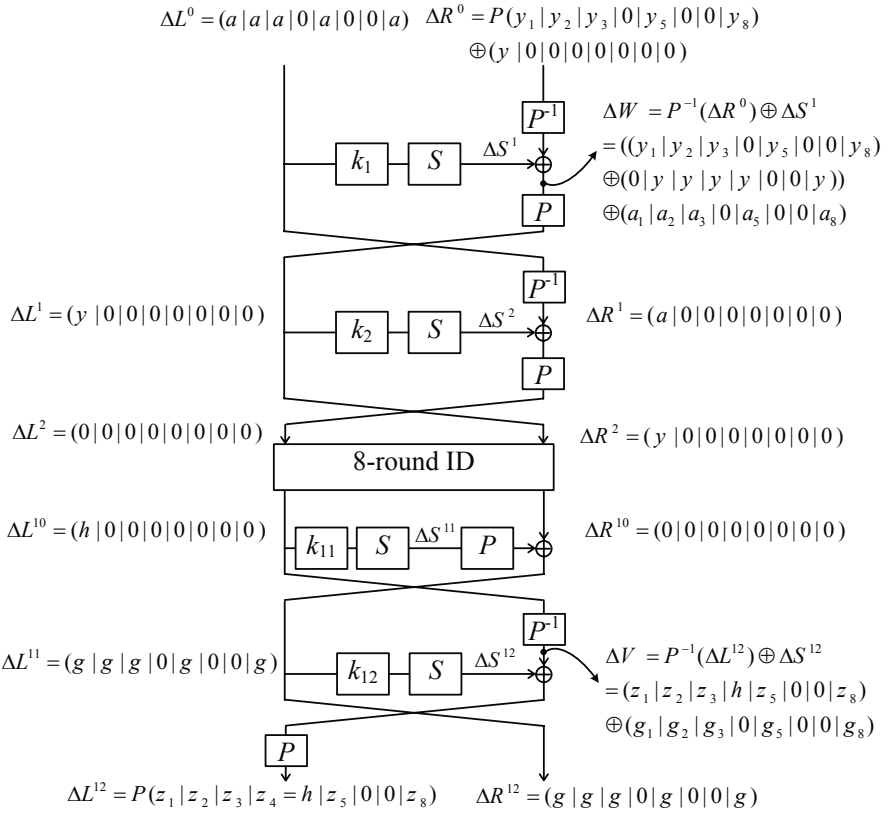

Fig. 4. 12–round impossible differential attack on reduced-round Camellia–128

It is obvious that each structure proposes about 2^{56} plaintexts, and 2^{111} plaintext pairs can be obtained from each structure. Totally, we can collect about 2^{n+56} plaintexts and 2^{n+111} plaintext pairs with the difference $\Delta L^0 = (a|a|a|0|a|0|0|a)$ and $\Delta R^0 = P(y_1|y_2|y_3|0|y_5|0|0|y_8) \oplus (y|0|0|0|0|0|0|0)$.

2. Obtain the ciphertexts of each structure and keep only the pairs that satisfy the following ciphertext difference:

$$\Delta L^{12} = P(z_1|z_2|z_3|h|z_5|0|0|z_8) \text{ and } \Delta R^{12} = (g|g|g|0|g|0|0|g)$$

where h, g and $z_\times$ are any non–zero byte values. The probability of this condition is $2^{-16} \times 2^{-56} = 2^{-72}$.

Thus the expected number of the remaining pairs is $2^{n+111} \times 2^{-72} = 2^{n+39}$.

3. Perform the following substeps:

   (a) Guess the 8–bit value of k_1^1 and partially encrypt every remaining plaintext pair to get ΔW_1 in the output of the XOR of Round 1 (see Fig. 4). Keep only the pairs whose ΔW_1 is zero. The probability of this event is 2^{-8}, thus we expect about $2^{n+39} \times 2^{-8} = 2^{n+31}$ pairs remain.

(b) For $l = 2, 3, 5, 8$ guess the 8–bit value of k_l^1 and partially encrypt every remaining plaintext pair to get ΔW_l. Keep only the pairs whose ΔW_l is equal to y (consider that y is already determined by ΔR^0 for each plaintext pair). The probability of this event for each l is 2^{-8}, thus the expected number of remaining pairs is $2^{n+31} \times 2^{-8\times 4} = 2^{n-1}$.

4. In this step consider the corresponding ciphertext pairs (C, C^*) of the remaining pairs then perform the following substeps:

(a) Guess the 8-bit value of k_1^{12}. Notice that according to Table 3, four bits of k_1^{12} is already fixed by k_1^1 previously guessed in step 3.a. Partially decrypt every remaining ciphertext pair (C, C^*) to get the first byte of the intermediate value ΔV in the output of the XOR of Round 12 (see Fig. 4). Keep only the pairs whose ΔV_1 is equal to zero. The probability of this condition is 2^{-8}, thus we expect about $2^{n-1} \times 2^{-8} = 2^{n-9}$ pairs remain.

(b) For $l = 2, 3$ obtain the 8–bit value of k_l^{12}. Notice that according to Table 3, all bits of k_l^{12} is already fixed by $k_{1,2,3}^1$ previously guessed in step 3. Partially decrypt every remaining ciphertext pair (C, C^*) to get the l–th byte of the intermediate value ΔV. Keep only the pairs whose ΔV_l is equal to h (consider that h is already determined by ΔL^{12} for each remaining ciphertext pair). The probability of this event for each l is 2^{-8}, thus the expected number of remaining pairs is $2^{n-9} \times 2^{-8\times 2} = 2^{n-25}$.

(c) For $l = 5, 8$ guess the 8–bit value of k_l^{12}. Notice that according to Table 3, four bits of k_5^{12} and four bits of k_8^{12} are already fixed by previously guessed k_5^1 and k_8^1, respectively. Partially decrypt every remaining ciphertext pair (C, C^*) to get the l–th byte of the intermediate value ΔV. Keep only the pairs whose ΔV_l is equal to h (consider that h is already determined by ΔL^{12} for each of the remaining ciphertext pairs). The probability of this event for each l is 2^{-8}, thus the expected number of remaining pairs is $2^{n-25} \times 2^{-8\times 2} = 2^{n-41}$.

(d) Guess the 24-bit value of $k_{4,6,7}^{12}$. Notice that according to Table 3, four bits of k_4^{12} and four bits of k_6^{12} are already fixed by previously guessed k_3^1 and k_5^1, respectively. Partially decrypt every remaining ciphertext pair (C, C^*) to get the exact value of intermediate pairs (L_1^{10}, L_1^{*10}). Consider that with probability 1, $\Delta V_{4,6,7} = h|0|0$. So this step does not affect the number of the remaining pairs.

5. Guess the 8-bit value of k_1^{11}. Notice that according to Table 3, four bits of k_1^{11} is already fixed by k_8^1 previously guessed in step 3. For every remaining pair, partially decrypt the (L_1^{10}, L_1^{*10}) through the first s-box of Round 11 to obtain ΔS_1^{11} and check if ΔS_1^{11} is equal to g, where g is already determined by ΔR^{12} for each ciphertext pair (see Fig. 4). The probability of this event is 2^{-8}, thus the expected number of remaining pairs is $2^{n-41} \times 2^{-8} = 2^{n-49}$.

In this stage of the attack, for every 72–bit guess of the subkeys $k_{1,2,3,5,8}^1, k_7^{12}$, 4 bits of $k_{1,4,5,6,8}^{12}$, and 4 bits of k_1^{11} we expect to obtain about 2^{n-49} pairs that satisfy the output difference of the 8–round impossible differential and also satisfy the difference $\Delta L_1 = \Delta R_2 = (y|0|0|0|0|0|0|0)$.

6. In this step, consider the corresponding plaintext pairs (P, P^*) of the remaining pairs then obtain the 24–bit value of $k^1_{4,6,7}$ (Notice that according to Table 3, all these 24 bits are already fixed by $k^{12}_{4,5,6,7,8}$). Now all bytes of k^1 are known, so partially encrypt every remaining pair to get the exact value of intermediate pairs (L^1_1, L^{*1}_1). Consider that with probability 1, $\Delta W_{4,6,7} = y|0|0$. So this step does not affect the number of the remaining pairs.
7. Guess the 8–bit value of k^2_1. Notice that according to Table 3, four bits of k^2_1 is already fixed by k^{11}_1 previously guessed in step 5. Then partially encrypt the (L^1_1, L^{*1}_1) through the first s–box of Round 2 to obtain ΔS^2_1 and check if ΔS^2_1 is equal to a, where a is already determined by ΔL^0 (see Fig. 4). If there exists a pair that passes this test, i.e. a pair that meets the input difference of the 8–round impossible differential, then discard the 76–bit subkey guess, and try another; otherwise for every 76-bit subkey guess, exhaustively search for the remaining 52 bits to recover the whole of k_A. Considering the relation between k_L and k_A, described in Section 3.2, this will lead to recovering the master key k_L.

3.4 Complexity of the Attack

In step 7, the probability that the difference ΔS^2_1 is equal to a fixed value a, is about 2^{-8}. So we expect only about $\epsilon = 2^{76}(1 - 2^{-8})^{2^{n-49}}$ guesses for 76–bit target subkey remain. If we accept the ϵ be equal to 1, then n will be 62.7. Thus the attack requires $2^{n+56} = 2^{118.7}$ chosen plaintexts.

In step 2, to get the qualified pairs, we first store the ciphertexts of each structure in a hash table indexed by the 4–th, 6–th and 7–th bytes of R^{12}, the XOR of the 1–st and 2–nd bytes of R^{12}, the XOR of the 1–st and 3–rd bytes of R^{12}, the XOR of the 1–st and 5–th bytes of R^{12}, the XOR of the 1–st and 8–th bytes of R^{12}, the 6–th and 7–th bytes of $P^{-1}(L^{12})$. Thus, every 2 ciphertexts with the same index in this table have the proper difference:

$$\Delta C = \Delta L^{12}|\Delta R^{12} = P(z_1|z_2|z_3|h|z_5|0|0|z_8)|(g|g|g|0|g|0|0|g).$$

Computing the 6–th and 7–th bytes of $P^{-1}(L^{12})$, requires 8 XOR operations, while each round of Camellia requires 24 XOR operations and 8 substitutions [2]. Thus, the total time complexity of computing the 6–th and 7–th bytes of $P^{-1}(L^{12})$ is less than about $8 \times 2^{118.7} \times \frac{1}{24} \times \frac{1}{12} \approx 2^{113.5}$ encryptions. Considering the complexity of obtaining the ciphertexts, step 2 requires about $2^{118.7}+2^{113.5} \approx 2^{118.7}$ encryptions. At the end of this step, we expect about $2^{n+39} = 2^{101.7}$ proper pairs to be accessible.

According to procedure described in section 3.3 the time complexity (in terms of encryption units) of steps 3–7 for recovering 76 bits of k_A is as follows:

$$\text{Step 3(a)} : 2 \times \frac{1}{8} \times \frac{1}{12} \times 2^{n+39} \times 2^8 = \frac{1}{12} \times 2^{n+45}$$

$$\text{Step 3(b)} : 2 \times \frac{1}{8} \times \frac{1}{12} \times \sum_{i=0}^{3} (2^{n+31-8i} \times 2^{8+8\times(i+1)}) = \frac{1}{12} \times 2^{n+47}$$

$$\text{Step 4(a)} : 2 \times \frac{1}{8} \times \frac{1}{12} \times 2^{n-1} \times 2^{40+4} = \frac{1}{12} \times 2^{n+41}$$

$$\text{Step 4(b)} : 2 \times \frac{1}{8} \times \frac{1}{12} \times \sum_{i=0}^{1} (2^{n-9-8i} \times 2^{44}) \approx \frac{1}{12} \times 2^{n+33}$$

$$\text{Step 4(c)} : 2 \times \frac{1}{8} \times \frac{1}{12} \times \sum_{i=0}^{1} (2^{n-25-8i} \times 2^{44+4\times(i+1)}) = \frac{1}{12} \times (2^{n+21} + 2^{n+19})$$

$$\text{Step 4(d)} : 2 \times \frac{3}{8} \times \frac{1}{12} \times 2^{n-41} \times 2^{52+4+4+8} = 2^{n+23}$$

$$\text{Step 5} : 2 \times \frac{1}{8} \times \frac{1}{12} \times 2^{n-41} \times 2^{68+4} = \frac{1}{12} \times 2^{n+29}$$

$$\text{Step 6} : 2 \times \frac{3}{8} \times \frac{1}{12} \times 2^{n-49} \times 2^{72+0} = 2^{n+19}$$

$$\text{Step 7} : 2 \times \frac{1}{8} \times \frac{1}{12} \times 2^{72+4} \times \sum_{i=0}^{2^{n-49}-1} (1 - 2^{-8})^i \approx \frac{1}{12} \times 2^{82} \times (1 - e^{-2^{n-57}})$$

Thus the dominant part of time complexity to recover 76 bits of k_A is related to steps 2, 3(a) and 3(b) which is about $\frac{1}{12} \times (2^{n+45} + 2^{n+47}) + 2^{118.7} \approx 2^{118.7}$ encryptions. In order to recover the whole of master key (k_L), for each of the 76-bit candidates (outputs of the procedure described in Section 3.3 which is expected to be about ϵ) we have to exhaustively search the remaining 52 bits of k_A. Then using the second result of Section 3.2, we can obtain k_L for each of these 52–bit guesses. As we described in Section 3.2, this operation requires about $\epsilon \times 4 \times 2^{52} \times \frac{1}{12}$ encryptions. Also one additional encryption is required to check the key with a plaintext/ciphertext pair. Finally, the overall time complexity to recover the master key is about $2^{118.7} + \epsilon \times 2^{52} \times (\frac{4}{12} + 1)$. For $\epsilon = 1$, the complexity will be about $2^{118.7}$ encryptions.

If we let the ϵ be about 2^{62}, then n will be equal to 60.3. Thus, data complexity of the proposed attack reduces to $2^{n+56} = 2^{116.3}$ and the dominant time complexity is composed of the time complexity of steps 2, 3(a), 3(b) and the exhaustive search in step 7, as below

$$2^{n+56} + \frac{1}{12} \times (2^{n+45} + 2^{n+47}) + \epsilon \times 2^{52} \times (\frac{4}{12} + 1) \approx 2^{116.6}.$$

4 Conclusion

In this paper, we proposed a new impossible differential attack on 12–round Camellia–128 without the FL/FL^{-1} functions. The attack uses a previously known 8–round impossible differential to retrieve the whole of the master key. The proposed attack exploits the redundancy in the key schedule of Camellia–128 to reduce the complexity. In this attack also we use the strategy of moving between the additional rounds in a zigzag path to accelerate the filtration of wrong pairs for each key guesses. Using these techniques along with a hash table to simplify the selection of proper pairs, the proposed attack requires about $2^{116.3}$

plaintexts, and has a time complexity equivalent to about $2^{116.6}$ encryptions. Our attack is the first successful impossible differential attack on 12 rounds of Camellia–128.

References

1. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: Camellia: a 128-bit Block Cipher Suitable for Multiple Platforms-Design and Analysis. In: Stinson, D.R., Tavares, S. (eds.) SAC 2000. LNCS, vol. 2012, pp. 39–56. Springer, Heidelberg (2001)
2. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: Specification of Camellia – a 128-bit Block Cipher. version 2.0 (2001), `http://info.isl.ntt.co.jp/crypt/eng/camellia/specifications.html`
3. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
4. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer, Heidelberg (1993)
5. CRYPTREC – Cryptography Research and Evaluation Committees, report, Archive (2002), `http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html`
6. Duo, L., Li, C., Feng, K.: Square Like Attack on Camellia. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 269–283. Springer, Heidelberg (2007)
7. 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. 129–146. Springer, Heidelberg (2003)
8. He, Y., Qing, S.: Square Attack on Reduced Camellia Cipher. In: Qing, S., Okamoto, T., Zhou, J. (eds.) ICICS 2001. LNCS, vol. 2229, pp. 238–245. Springer, Heidelberg (2001)
9. International Standardization of Organization (ISO), International Standard - ISO/IEC 18033-3, Information technology - Security techniques - Encryption algorithms - Part 3: Block ciphers (July 2005)
10. 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)
11. Kawabata, T., Kaneko, T.: A Study on Higher Order Differential Attack of Camellia. In: The 2nd open NESSIE workshop (2001)
12. Knudsen, L.R.: DEAL – a 128-bit Block Cipher. Technical report, Department of Informatics, University of Bergen, Norway (1998)
13. Lee, S., Hong, S., Lee, S., Lim, J., Yoon, S.: Truncated Differential Cryptanalysis of Camellia. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 32–38. Springer, Heidelberg (2002)
14. Lei, D., Chao, L., Feng, K.: New Observation on Camellia. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 51–64. Springer, Heidelberg (2006)
15. Lu, J.: Cryptanalysis of Block Ciphers. PhD Thesis, Department of Mathematics, Royal Holloway, University of London, England (2008)
16. Lu, J., Kim, J., Keller, N., Dunkelman, O.: Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 370–386. Springer, Heidelberg (2008)

17. NESSIE – New European Schemes for Signatures, Integrity, and Encryption, final report of European project IST-1999-12324. Archive (1999), https://www.cosic.esat.kuleuven.be/nessie/Bookv015.pdf
18. NTT Information Sharing Platform Laboratories: Internationally Standardized Encryption Algorithm from Japan"Camellia", http://info.isl.ntt.co.jp/crypt/camellia/dl/Camellia20061108v4_eng.pdf
19. Shirai, T.: Differential, Linear, Boomerang and Rectangle Cryptanalysis of Reduced-Round Camellia. In: Proceedings of 3rd NESSIE workshop (November 2002)
20. Sugita, M., Kobara, K., 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)
21. Wu, W., Feng, D., Chen, H.: Collision Attack and Pseudorandomness of Reduced-Round Camellia. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 252–266. Springer, Heidelberg (2004)
22. Wu, W., Zhang, L., 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)
23. Wu, W., Zhang, W., Feng, D.: Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. Journal of Computer Science and Technology 22(3), 449–456 (2007)
24. Yeom, Y., Park, S., 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)