

Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1

Jiqiang Lu^{1*}, Jongsung Kim^{2**}, Nathan Keller^{3***}, and Orr Dunkelman^{4†}

¹ Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK

lvjiqiang@hotmail.com

² Center for Information Security Technologies(CIST), Korea University
Anam Dong, Sungbuk Gu, Seoul, Korea

joshep@cist.korea.ac.kr

³Einstein Institute of Mathematics, Hebrew University
Jerusalem 91904, Israel

nkeller@math.huji.ac.il

⁴ESAT/SCD-COSIC, Katholieke Universiteit Leuven
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium.

orr.dunkelman@esat.kuleuven.be

Abstract. Camellia and MISTY1 are Feistel block ciphers. In this paper, we observe that, when conducting impossible differential cryptanalysis on Camellia and MISTY1, their round structures allow us to partially determine whether a candidate pair is right by guessing only a small fraction of the unknown required subkey bits of a relevant round at a time, instead of all of them. This reduces the computational complexity of an attack, and may allow us to break more rounds of a cipher. Taking advantage of this main observation, we significantly improve previous impossible differential cryptanalysis on reduced Camellia and MISTY1, obtaining the best published cryptanalytic results against both the ciphers.

Key words: Block cipher cryptanalysis, Camellia, MISTY1, Impossible differential cryptanalysis

* This author as well as his work was supported by a British Chevening / Royal Holloway Scholarship and the European Commission under contract IST-2002-507932 (ECRYPT).

** This author was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2006-(C1090-0603-0025)).

*** This author was supported by the Adams fellowship.

† This work was supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government and by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy).

1 Introduction

Camellia [1] is a 128-bit Feistel block cipher with a user key length of 128, 192 or 256 bits, while MISTY1 [18] is a 64-bit Feistel block cipher with a 128-bit user key. Both Camellia and MISTY1 were selected to be CRYPTREC [6] e-government recommended ciphers in 2002 and NESSIE [19] block cipher selections in 2003, and were adopted as ISO [9] international standards in 2005. Since Camellia and MISTY1 are increasingly being used in many real-life cryptographic applications, it is essential to continuing to investigate their security against different cryptanalytic attack scenarios. For simplicity, we denote the three versions of Camellia by Camellia-128/192/256, respectively.

Many cryptanalytic results on Camellia and MISTY1 have been published [2,7,10,11,14,16,21,22,23,24,25,26]. In summary, the best cryptanalytic results on Camellia without the FL functions are the truncated differential cryptanalysis [12] on 8-round Camellia-128 [15], the impossible differential cryptanalysis on 12-round Camellia-192 [25], and the linear [17] and impossible differential cryptanalysis on 12-round Camellia-256 [21,25]; the best cryptanalytic result on MISTY1 without the FL functions is the impossible differential cryptanalysis on 6-round MISTY1 [10].

Impossible differential cryptanalysis [3,13], as a special case of differential cryptanalysis [5], uses one or more differentials with a zero probability, called impossible differentials, which are usually built in a miss-in-the-middle manner [4]. In the impossible differential attacks on Camellia and MISTY1 described in [10,25], the general approach is to guess all the unknown required subkey bits of a relevant round to partially decrypt (or encrypt) a candidate pair through the round function; finally one checks whether the pair could produce the expected difference just before (resp. after) the round.

In this paper, however, we observe that, due to the round structures of Camellia and MISTY1, we can partially check whether a candidate pair could produce the expected difference by guessing only a small fraction of the unknown required round subkey bits at a time, and do a series of partial checks by guessing other fractions of the unknown required subkey bits, instead of guessing all the unknown required subkey bits at once. Since some wrong pairs can be discarded before the next guess for a different fraction of the required round subkey bits, we can reduce the computational workload for an attack, and even more importantly, we may break more rounds of a cipher. We call this the early abort technique¹. Taking advantage of the early abort technique, we improve the previous impossible differential attacks on 12-round Camellia-192 without the FL functions and 6-round MISTY1 without the FL functions, and present impossible differential cryptanalysis of 11-round Camellia-128 without the FL functions and 13-round Camellia-256 without the FL functions, following [10,25]. These are the best published cryptanalytic results on Camellia and MISTY1. Table 1 compares our new results with those previously known on Camellia and MISTY1, where “none” means that the relevant block cipher has no FL function, “all”

¹ Note that this name is also used for other attacks in some previous works.

Table 1. Comparison of our new cryptanalytic results with those previously known on Camellia and MISTY1

Cipher	Type of Attack	Rounds	FL/FL ⁻¹	Data	Time	Source
Camellia-128 (18 rounds)	Square attack	6	none	2 ^{11.7} CP	2 ¹¹²	[8]
	Truncated differential	8	none	2 ^{83.6} CP	2 ^{55.6}	[15]
	Impossible differential	7	none	—	—	[22]
		11	none	2 ¹²⁰ CP	2 ^{83.4}	Sect. 4
Camellia-192/256 (24 rounds)	Boomerang attack	9	all	2 ¹²⁴ ACPC	2 ¹⁷⁰	[21]
	Collision attack	9	none	2 ¹³ CP	2 ^{175.6}	[24]
	Square attack	10	none	—	2 ¹⁸⁶	[16]
	Impossible differential	12	none	2 ¹²⁰ CP	2 ¹⁸¹	[25]
		12	none	2 ¹¹⁹ CP	2 ^{147.3}	Sect. 4
Camellia-256 (24 rounds)	Square attack	9	all	2 ⁶⁰ CP	2 ²⁰²	[26]
	Integral cryptanalysis	9	all	2 ^{60.5} CP	2 ^{202.2}	[27]
	Rectangle attack	10	all	2 ¹²⁷ CP	2 ²⁴¹	[21]
	Collision attack	10	none	2 ¹⁴ CP	2 ^{239.9}	[24]
	Differential	11	none	2 ¹⁰⁴ CP	2 ²³²	[21]
	High-order differential	11	none	2 ²¹ CP	2 ²⁵⁵	[7]
		11	all	2 ⁹³ CP	2 ²⁵⁶	[7]
	Square attack	11	none	—	2 ²⁵⁰	[16]
	Linear cryptanalysis	12	none	2 ¹¹⁹ CP	2 ²⁴⁷	[21]
	Impossible differential	13	none	2 ¹²⁰ CP	2 ^{211.7}	Sect. 4
MISTY1 (8 rounds)	High-order differential	5	none	2 ^{10.5} CP	2 ¹⁷	[23]
	Integral cryptanalysis	5	most	2 ³⁴ CP	2 ⁴⁸	[14]
	Slicing attack	4	all	2 ^{22.25} CP	2 ⁴⁵	[11]
	Slicing attack+impossible	4	all	2 ^{27.2} CP	2 ^{81.6}	[11]
		4	all	2 ^{27.5} CP	2 ¹¹⁶	[11]
	Impossible differential	6	none	2 ⁵⁴ CP	2 ⁶¹	[10]
		6	none	2 ³⁹ CP	2 ¹⁰⁶	[10]
		6	none	2 ³⁹ CP	2 ⁸⁵	Sect. 5

CP: Chosen Plaintexts, ACPC: Adaptive Chosen Plaintexts and Ciphertexts
Time unit: Encryptions

means that it has all the FL functions, and “most” means that it has all the FL functions except the ones in the final swap layer.

The rest of the paper is organised as follows. In the next section, we briefly describe Camellia and MISTY1. In Section 3, we introduce the early abort technique in a general way. In Sections 4 and 5, we present our cryptanalytic results on Camellia and MISTY1, respectively. Section 6 summaries this paper.

2 Preliminaries

Throughout the paper, we denote the bit-wise exclusive OR (XOR) operation by \oplus , and bit string concatenation by \parallel .

2.1 The Camellia Block Cipher

Camellia [1] takes a 128-bit plaintext P as input, and has a total of N rounds, where N is 18 for Camellia-128, and 24 for Camellia-192/256. Its encryption procedure is as follows.

1. $L^0 || R^0 = P \oplus (KW_1 || KW_2)$
2. For $i = 1$ to N :
 - if $i = 6$ or 12 (or 18 for Camellia-192/256),
 $L^i = \text{FL}(L^{i-1}, KI_{i/3-1}), R^i = \text{FL}^{-1}(R^{i-1}, KI_{i/3});$
 - else
 $L^i = \text{F}(L^{i-1}, K_i) \oplus R^{i-1}, R^i = L^{i-1};$
3. Ciphertext $C = (R^N \oplus KW_3) || (L^N \oplus KW_4),$

where KW , K and KI are 64-bit round subkeys, L^i and R^i are 64 bits long, and the F function comprises a XOR operation, then an application of 8 parallel nonlinear 8×8 -bit bijective S-boxes s_1, s_2, \dots, s_8 , and, finally, a linear P function. As we consider the version of Camellia without the FL functions, we omit the description of the two functions FL and FL^{-1} ; see [1] for their specifications. The P function and its inverse P^{-1} are defined over $GF(2^8)^8 \rightarrow GF(2^8)^8$, as follows.

$$\text{P} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}, \text{P}^{-1} = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

2.2 The MISTY1 Block Cipher

MISTY1 [18] takes a 64-bit plaintext P as input, and has a total of 8 rounds. Its encryption procedure is as follows.

1. $P = L^0 || R^0, KL = KL_1 || KL_2 || \dots || KL_{10}, KI = KI_1 || KI_2 || \dots || KI_8,$
 $KO = KO_1 || KO_2 || \dots || KO_8.$
2. For $i = 1, 3, 5, 7$:
 - $R^i = \text{FL}(L^{i-1}, KL_i), L^i = \text{FL}(R^{i-1}, KL_{i+1}) \oplus \text{FO}(R^i, KO_i, KI_i),$
 - $L^{i+1} = R^i \oplus \text{FO}(L^i, KO_{i+1}, KI_{i+1}), R^{i+1} = L^i.$
3. Ciphertext $C = \text{FL}(R^8, KL_{10}) || \text{FL}(L^8, KL_9),$

where KL , KI and KO are round subkeys, and the function FO takes as inputs a 32-bit block X and two 32-bit subkeys KO_i and KI_i , and outputs a 32-bit block Y , and is defined as follows (see [18] for details of the FL function).

1. $X = XL_0 || XR_0, KO_i = KO_{i1} || KO_{i2} || KO_{i3} || KO_{i4}, KI_i = KI_{i1} || KI_{i2} || KI_{i3}.$
2. For $j = 1, 2, 3$:
 - $XR_j = \text{FI}(XL_{j-1} \oplus KO_{ij}, KI_{ij}) \oplus XR_{j-1}, XL_j = XR_{j-1}.$
3. $Y = (XL_3 \oplus KO_{i4}) || XR_3.$

In the above description, the FI function takes a 16-bit block X and a subkey KI_{ij} as inputs, and outputs a 16-bit block Y , computed as defined below.

1. $X = XL_0(9 \text{ bits}) || XR_0(7 \text{ bits})$, $KI_{ij} = KI_{ijL}(7 \text{ bits}) || KI_{ijR}(9 \text{ bits})$,
2. $XL_1 = XR_0$, $XR_1 = S_9(XL_0) \oplus \text{Extnd}(XR_0)$,
3. $XL_2 = XR_1 \oplus KI_{ijR}$, $XR_2 = S_7(XL_1) \oplus \text{Trunc}(XR_1) \oplus KI_{ijL}$,
4. $XL_3 = XR_2$, $XR_3 = S_9(XL_2) \oplus \text{Extnd}(XR_2)$,
5. $Y = XL_3 || XR_3$,

where S_9 is a 9×9 -bit bijective S-box, S_7 is a 7×7 -bit bijective S-box, the function Extnd extends from 7 bits to 9 bits by concatenating two zeros on the left side, and the function Trunc truncates two bits from the left side.

3 A General Description of the Early Abort Technique

Impossible differential cryptanalysis is based on one or more impossible differentials, written $\alpha \nrightarrow \beta$, and it usually treats a block cipher $E : \{0, 1\}^n \times \{0, 1\}^k \rightarrow \{0, 1\}^n$ as a cascade of three sub-ciphers $E = E_a \circ E_0 \circ E_b$, where E_0 denotes the rounds for which $\alpha \nrightarrow \beta$ holds, E_b denotes a few rounds before E_0 , and E_a denotes a few rounds after E_0 . Given a guess for the subkeys used in E_b and E_a , if a plaintext pair produces a difference of α just after E_b , and its corresponding ciphertext pair produces a difference of β just before E_a , then this guess for the subkey must be incorrect. Thus, given a sufficient number of matching plaintext/ciphertext pairs, we can find the correct subkey by discarding the wrong guesses.

When checking if a plaintext pair produces a difference of α just after E_b (or its corresponding ciphertext pair produces a difference of β just before E_a), the general approach is to guess all the unknown bits of the relevant round subkey necessary to partially encrypt (resp. decrypt) the pair through the substitution and diffusion layers; finally, one can check whether the pair could produce an expected difference just after (resp. before) the round. To make matters more specific, consider a general Feistel structure as in Camellia; as shown in Fig. 1, we assume it has a nonlinear substitution consisting of m parallel S-boxes and a linear diffusion function P . For simplicity, we assume the round in Fig. 1 is just before E_0 ; that is to say, the attacker is looking for a pair with difference $(\Delta L_{i+1} || \Delta R_{i+1}) = \alpha$. According to previous attack procedures, due to the diffusion of the P function, the attacker will guess all the required unknown subkey bits (i.e. those corresponding to the active S-boxes) at a time, then encrypt the left halves of the pair through the substitution layer to get the difference just after the P function, and finally XOR it with the difference ΔR_i to check if it has the difference α after the round.

However, we observe that the round structure can allow us to partially determine whether a candidate pair could produce the expected difference α by guessing only a small fraction of the required round subkey bits at a time, instead of all of them simultaneously. More specifically, since we know the expected difference α and the intermediate values of the pair just before the round, we

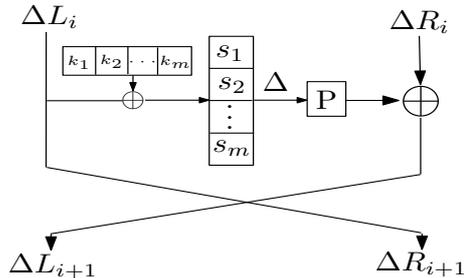


Fig. 1. A general Feistel structure

can compute the expected difference just before the P function, denoted by $\Delta (= P^{-1}(\Delta R_i \oplus \Delta L_{i+1}))$, as the P function is usually linearly invertible. Only if the expected difference Δ appears after the substitution layer could the pair produce the difference α after the round. Thus, in the following, we guess only those of the required unknown subkey bits corresponding to one (or more) active S-box, then encrypt the pair through the S-box, and finally check if it produces the corresponding partial difference in Δ . If not, then the pair is not right, and we can discard it immediately; otherwise, we guess another part of the required round subkey bits corresponding to another active S-box, and check the pair similarly. A pair is right only if it could produce the partial difference out of the expected difference Δ just before the P function, under every part of the required round subkey bits. Some wrong pairs can be discarded before the next guess; by this observation we can reduce the computational workload of an attack, and even more significantly, we may break more rounds.

4 Impossible Differential Cryptanalysis of Reduced Camellia

As Camellia is byte-oriented, we represent the 128 bits before (or after) a round as 16 bytes, and denote the l -th byte of a subkey K_i by $k_{i,l}$, ($1 \leq l \leq 8$).

Most recently, Wu et al. [25] presented an impossible differential cryptanalysis of 12-round Camellia-192/256 without the FL functions, which is based on the following 8-round impossible differentials: $(0, 0, 0, 0, 0, 0, 0, 0, a, 0, 0, 0, 0, 0, 0, 0) \rightarrow (h, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$, where a and h are any two nonzero bytes. See Fig. 2 for details of these 8-round impossible differentials.

In this section, we present an impossible differential cryptanalysis on 13-round Camellia-256, 12-round Camellia-192 and 11-round Camellia-128, following the work described in [25]. These are the best published cryptanalytic results on Camellia without the FL functions.

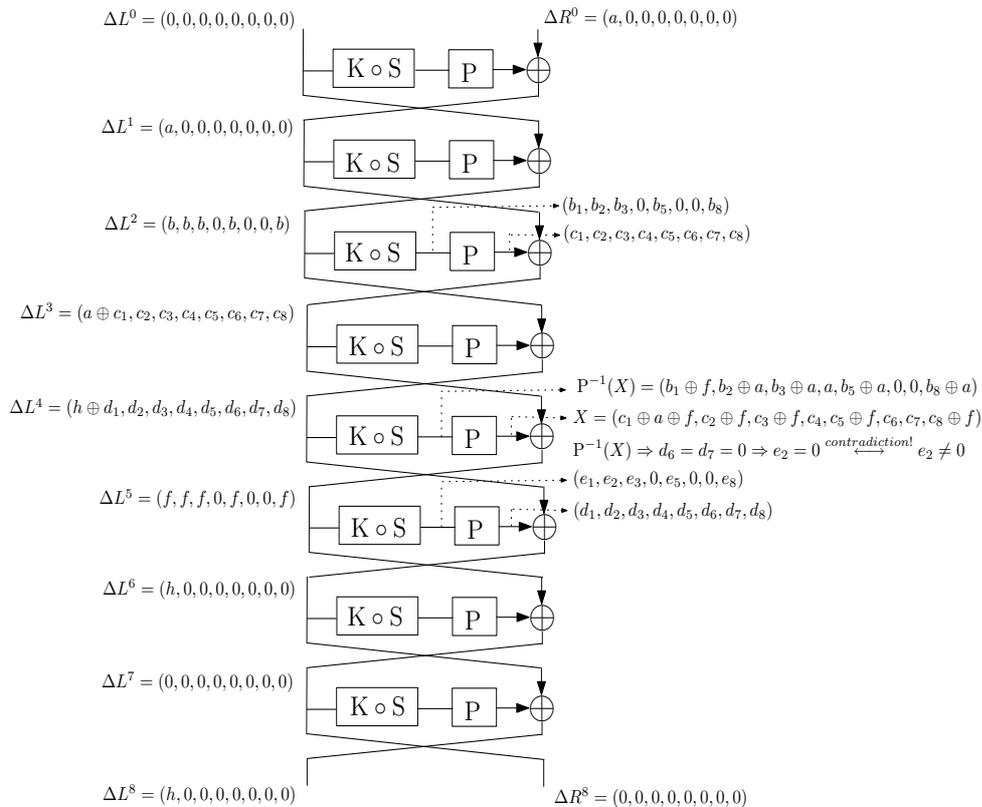


Fig. 2. 8-round impossible differentials of Camellia

4.1 Attacking 13-Round Camellia-256

We use the 8-round impossible differentials in Rounds 4 to 11. As every S-box has a minimal nonzero probability of 2^{-7} , an output difference $(h, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ of the 8-round impossible differentials propagates to at most 2^7 possible output differences $(g, g, g, 0, g, 0, 0, g, h, 0, 0, 0, 0, 0, 0, 0)$ after Round 12, where g is nonzero. Then, every $(g, g, g, 0, g, 0, 0, g, h, 0, 0, 0, 0, 0, 0, 0)$ propagates to at most $(2^7)^5$ possible output differences after Round 13. Hence, there are at most $(2^8 - 1) \cdot 2^7 \cdot (2^7)^5 \approx 2^{50}$ possible output differences after Round 13; we denote them by the set Δ_{13} .

We can use the early abort technique in the first two rounds and the last round of the 13-round attack. Consider a plaintext pair $(P_i = (L_i^0, R_i^0), P_j = (L_j^0, R_j^0))$ with an output difference $(\Delta L^{13}, \Delta R^{13})$ belonging to Δ_{13} . The difference just after the S-box substitution layer of Round 13 must have the form $(?, ?, ?, 0, ?, 0, 0, ?)$, where the question mark $?$ denotes an unknown byte difference (two bytes marked with $?$ may be different); and there must be a h such that

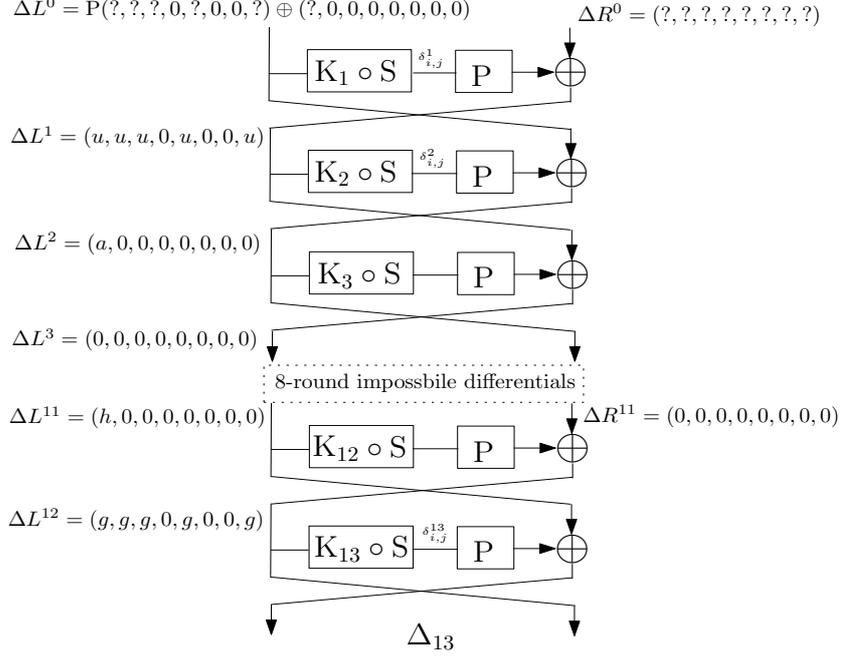


Fig. 3. Impossible differential attack on 13-round Camellia-256

$P^{-1}(L_i^{13} \oplus L_j^{13} \oplus (h, 0, 0, 0, 0, 0, 0, 0))$ has the form $(?, ?, ?, 0, ?, 0, 0, ?)$. h has 255 possible values, but only one of them satisfies the above condition. The reason is as follows. If we assume there are two different values h_1 and h_2 that satisfy the condition, then observe that $P^{-1}((h_1, 0, 0, 0, 0, 0, 0, 0) \oplus (h_2, 0, 0, 0, 0, 0, 0, 0))$ also has the form $(?, ?, ?, 0, ?, 0, 0, ?)$; note that the 4-th byte is 0; however, by the P^{-1} function we know the 4-th byte should be $h_1 \oplus h_2 \neq 0$. This gives a contradiction. On the other hand, $P^{-1}(R_i^0 \oplus R_j^0 \oplus (u, u, u, 0, u, 0, 0, u))$ has a unique value in the first two bytes for every nonzero value of u , because, if we suppose that there are two values u_1 and u_2 such that $P^{-1}(R_i^0 \oplus R_j^0 \oplus (u_1, u_1, u_1, 0, u_1, 0, 0, u_1)) \oplus P^{-1}(R_i^0 \oplus R_j^0 \oplus (u_2, u_2, u_2, 0, u_2, 0, 0, u_2)) = (0, 0, ?, ?, ?, ?, ?, ?)$, then we get $P^{-1}(u_1 \oplus u_2, u_1 \oplus u_2, u_1 \oplus u_2, 0, u_1 \oplus u_2, 0, 0, u_1 \oplus u_2) = (0, 0, ?, ?, ?, ?, ?, ?)$; by the P^{-1} function we know that the first byte should be $u_1 \oplus u_2$, meaning that $u_1 = u_2$.

The above analysis enables us to give the following procedure for attacking 13-round Camellia-256. Fig. 3 illustrates the attack.

1. Choose 2^8 structures: a structure is a set of 2^{112} plaintexts $P_i = (L_i^0, R_i^0)$, with $L_i^0 = P(x_1, x_2, x_3, \alpha_4, x_5, \alpha_6, \alpha_7, x_8) \oplus (x, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8)$ and $R_i^0 = (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)$, where x, x_l and y_l take all the possible values in $GF(2^8)$, and α_l and β_l are fixed to certain values in $GF(2^8)$, ($i = 1, 2, \dots, 2^{112}$). In a chosen-plaintext attack scenario, obtain all their

- ciphertexts; we denote them by $C_i = (L_i^{13}, R_i^{13})$, respectively. For different values of $(x_1, x_2, x_3, x_5, x_8, x, y_1, \dots, y_8)$, the resultant 128-bit blocks are different; thus, there are $2^{112 \times 2} / 2 = 2^{223}$ plaintext pairs (P_i, P_j) in a structure ($j = 1, 2, \dots, 2^{112}$), so the 2^8 structures yield a total of 2^{231} ciphertext pairs. Keep only the pairs (C_i, C_j) with a difference belonging to Δ_{13} . The expected number of the remaining pairs is about $2^{231} \cdot \frac{2^{50}}{2^{128}} = 2^{153}$.
2. For every remaining ciphertext pair (C_i, C_j) , compute $P^{-1}(L_i^{13} \oplus L_j^{13} \oplus (h, 0, 0, 0, 0, 0, 0, 0))$ for all the 255 possible nonzero values of h . As discussed earlier there is only one value of h such that $P^{-1}(L_i^{13} \oplus L_j^{13} \oplus (h, 0, 0, 0, 0, 0, 0, 0))$ has the form $(?, ?, ?, 0, ?, 0, 0, ?)$; we denote by $\delta_{i,j}^{13}$ the value $P^{-1}(L_i^{13} \oplus L_j^{13} \oplus (h, 0, 0, 0, 0, 0, 0, 0))$ with the form $(?, ?, ?, 0, ?, 0, 0, ?)$. Then, do as follows.
 - (a) For $l = 1, 2, 3, 5, 8$:
 - Guess the byte $k_{13,l}$ of the subkey K_{13} ;
 - For every remaining pair (C_i, C_j) , partially decrypt the l -th byte of (R_i^{13}, R_j^{13}) through the s_l S-box, and check if they have a difference equal to the corresponding one-byte difference in $\delta_{i,j}^{13}$; keep only the qualified pairs. A proportion of about $1 - 2^{-7}$ of the remaining pairs will be discarded after every iteration.
 - (b) Guess the three bytes $(k_{13,4}, k_{13,6}, k_{13,7})$ of the subkey K_{13} , such that we can get the intermediate values just after Round 12 for every remaining pair.
 3. Guess the byte $k_{12,1}$ of the subkey K_{12} . For every remaining ciphertext pair (C_i, C_j) , compute $s_1(R_{i,1}^{12} \oplus k_{12,1})$ and $s_1(R_{j,1}^{12} \oplus k_{12,1})$, and check if they have a difference equal to $L_{i,1}^{12} \oplus L_{j,1}^{12}$. Keep only the qualified pairs. The expected number of the remaining pairs is about $2^{118} \cdot 2^{-7} = 2^{111}$.
 4. For every plaintext pair (P_i, P_j) corresponding to a remaining ciphertext pair (C_i, C_j) , compute $P^{-1}(R_i^0 \oplus R_j^0 \oplus (u, u, u, 0, u, 0, 0, u))$ for all the 255 possible nonzero values of u ; we denote the values by $\Delta_{i,j}^1$, respectively. Then, do as follows.
 - (a) Guess the two bytes $(k_{1,1}, k_{1,2})$ of the subkey K_1 . For every plaintext pair (P_i, P_j) , partially encrypt the first two bytes of (L_i^0, L_j^0) through the s_1 and s_2 S-boxes, and check if they have a difference equal to any of the corresponding two-byte partial differences in $\Delta_{i,j}^1$. Keep only the qualified pairs; as discussed earlier there is only one difference in $\Delta_{i,j}^1$ for a qualified pair, and we denote this difference from $\Delta_{i,j}^1$ by $\delta_{i,j}^1$. As there are 255 possible values in $\Delta_{i,j}^1$ for every pair, the expected number of the remaining pairs is about $2^{111} \cdot \frac{255}{2^{16}} \approx 2^{103}$.
 - (b) For $l = 3$ to 8:
 - Guess the byte $k_{1,l}$ of K_1 ;
 - For every remaining pair (P_i, P_j) , partially encrypt the l -th byte of (L_i^0, L_j^0) through the s_l S-box, and check if they have a difference equal to the corresponding one-byte partial difference in $\delta_{i,j}^1$; keep only the qualified pairs. The difference $\delta_{i,j}^1$ is already fixed in Step 4-(a), so it is expected that a proportion of about $1 - 2^{-8}$ of the remaining pairs will be discarded after every iteration.

5. For every remaining plaintext pair (P_i, P_j) , compute $P^{-1}(L_i^0 \oplus L_j^0 \oplus (a, 0, 0, 0, 0, 0, 0, 0))$ for all the 255 possible nonzero values of a . Similarly, we know there is only one value of a such that $P^{-1}(L_i^0 \oplus L_j^0 \oplus (a, 0, 0, 0, 0, 0, 0, 0))$ has the form $(?, ?, ?, 0, ?, 0, 0, ?)$; we denote by $\delta_{i,j}^2$ the value $P^{-1}(L_i^0 \oplus L_j^0 \oplus (a, 0, 0, 0, 0, 0, 0, 0))$ with the form $(?, ?, ?, 0, ?, 0, 0, ?)$. Then, do as follows.
 - (a) For $l = 1, 2, 3, 5, 8$:
 - Guess the byte $k_{2,l}$ of the subkey K_2 ;
 - For every remaining pair (P_i, P_j) , partially encrypt the l -th byte of (L_i^1, L_j^1) through the s_l S-box, and check if they have a difference equal to the corresponding one-byte partial difference in $\delta_{i,j}^2$; keep only the qualified pairs. Similarly, it is expected that a proportion of about $1 - 2^{-8}$ of the remaining pairs will be discarded after every iteration.
 - (b) Guess the three bytes $(k_{2,4}, k_{2,6}, k_{2,7})$ of K_2 , so that we can get the intermediate values just after Round 2 of every remaining pair.
6. Guess the byte $k_{3,1}$ of the subkey K_3 . For every plaintext pair (P_i, P_j) , partially encrypt the first bytes of (L_i^2, L_j^2) through the s_1 S-box, and check if they have a difference equal to $L_{i,1}^1 \oplus L_{j,1}^1$. If there exists a plaintext pair that passes this test, then discard this subkey guess, and try another; otherwise, record the subkey guess. The expected number of the remaining subkey guesses is about $2^{208} \cdot (1 - 2^{-8})^{2^{15}} \approx 2^{208} \cdot e^{-2^8} \approx 2^{24}$.
7. For every recorded subkey guess (K_1, K_2) , exhaustively search for the remaining 128 key bits.

In Step 1, choosing the qualified pairs requires about $2^{120} \cdot \frac{120}{32} = 2^{121.9}$ memory accesses if conducted on a 32-bit computer; actually, it can be done more efficiently using computers of today. Step 2 has a time complexity of about $\sum_{i=0}^4 (2 \cdot 2^{153-7 \cdot i} \cdot 2^{8 \cdot (i+1)} \cdot \frac{1}{13} \cdot \frac{1}{8}) + 2 \cdot 2^{118} \cdot 2^{64} \cdot \frac{1}{13} \cdot \frac{3}{8} \approx 2^{177.9}$ decryptions. Step 3 has a time complexity of about $2 \cdot 2^{118} \cdot 2^{72} \cdot \frac{1}{13} \cdot \frac{1}{8} \approx 2^{184.3}$ decryptions. Step 4 has a time complexity of about $2 \cdot 2^{111} \cdot 2^{88} \cdot \frac{1}{13} \cdot \frac{2}{8} + \sum_{i=0}^5 (2 \cdot 2^{103-8 \cdot i} \cdot 2^{88+8 \cdot (i+1)} \cdot \frac{1}{13} \cdot \frac{1}{8}) \approx 2^{196.3}$ encryptions. Step 5 has a time complexity of about $\sum_{i=0}^4 (2 \cdot 2^{55-8 \cdot i} \cdot 2^{136+8 \cdot (i+1)} \cdot \frac{1}{13} \cdot \frac{1}{8}) + 2 \cdot 2^{15} \cdot 2^{200} \cdot \frac{1}{13} \cdot \frac{3}{8} \approx 2^{210.9}$ encryptions. Step 6 has a time complexity of about $2 \cdot 2^{208} \cdot [1 + (1 - 2^{-8}) + \dots + (1 - 2^{-8})^{2^{15}}] \cdot \frac{1}{13} \cdot \frac{1}{8} \approx 2^{210.3}$ encryptions. It is expected that Step 7 requires $2^{24} \cdot 2^{128} = 2^{152}$ trial encryptions to find the correct 256 key bits. Therefore, the attack has a total time complexity of about $2^{211.7}$ 13-round Camellia-256 computations.

4.2 Attacking 12-Round Camellia-192

The impossible differential cryptanalysis of 12-round Camellia-192 due to Wu et al. [25] can be improved in the following several ways.

- Take 2^7 structures in Step 2 (of Wu et al.'s attack). Thus, the expected number of the remaining plaintext pairs is about 2^{118} . As a result, the time complexity of Step 3 is about $2 \cdot 2^{118} \cdot 2^8 \cdot \frac{1}{12} \cdot \frac{1}{8} \approx 2^{120.4}$ 12-round Camellia-192 computations.

- Use the early abort technique for Round 1 in Step 4, just as we do in Step 4 of the attack in Section 4.1. This Step has a time complexity of about $2 \cdot 2^{110} \cdot 2^{24} \cdot \frac{1}{12} \cdot \frac{2}{8} + \sum_{i=0}^5 (2 \cdot 2^{102-8 \cdot i} \cdot 2^{24+8 \cdot (i+1)} \cdot \frac{1}{12} \cdot \frac{1}{8}) \approx 2^{131.4}$ 12-round Camellia-192 computations.
- Use the early abort technique for Round 2 in Step 5, just as we do in Step 5 of the attack in Section 4.1. This Step has a time complexity of about $\sum_{i=0}^4 (2 \cdot 2^{54-8 \cdot i} \cdot 2^{72+8 \cdot (i+1)} \cdot \frac{1}{12} \cdot \frac{1}{8}) + 2 \cdot 2^{14} \cdot 2^{136} \cdot \frac{1}{12} \cdot \frac{3}{8} \approx 2^{146}$ 12-round Camellia-192 computations.
- In Step 6, it is expected that about $2^{144} \cdot (1 - 2^{-8})^{2^{14}} \approx 2^{52}$ guesses for $(K_1, K_2, k_{3,1}, k_{12,1})$ remain. Then, for every remaining guess for (K_1, K_2) , exhaustively search for the remaining 64 key bits, which is expected to require 2^{116} trial encryptions to find the 192 key bits. This step has a time complexity of about $2 \cdot 2^{144} \cdot [1 + (1 - 2^{-8}) + \dots + (1 - 2^{-8})^{2^{14}}] \cdot \frac{1}{12} \cdot \frac{1}{8} + 2^{116} \approx 2^{146.4}$ 12-round Camellia-192 computations.

Therefore, the attack requires 2^{119} chosen plaintexts, and has a total time complexity of about $2^{147.3}$ 12-round Camellia-192 computations, dramatically lower than the time complexity of 2^{181} for Wu et al.'s attack.

4.3 Attacking 11-Round Camellia-128

To attack 11-round Camellia-128, we use the 8-round impossible differentials in Rounds 3 to 10, and use the early abort technique in the first round. The attack requires 2^{120} chosen plaintexts, and has a time complexity of about $2^{83.4} (\approx 2 \cdot 2^{71} \cdot 2^{15} \cdot \frac{1}{11} \cdot \frac{3}{8} + 2 \cdot 2^{80} \cdot [1 + (1 - 2^{-8}) + \dots + (1 - 2^{-8})^{2^{14}}] \cdot \frac{1}{11} \cdot \frac{1}{8} + 2^{64})$ 11-round Camellia-128 computations.

5 Impossible Differential Cryptanalysis of 6-Round MISTY1 without FL Functions

In 2001, Kühn [10] presented an impossible differential cryptanalysis on 6-round MISTY1 (without the FL functions); the attack requires 2^{39} plaintexts and has a time complexity of 2^{106} 6-round MISTY1 computations. Kühn also presented another impossible differential cryptanalysis on 6-round MISTY1, which requires more plaintexts but less computations. Both the attacks are based on the following generic 5-round impossible differentials for Feistel networks with bijective round structures: $(0, 0, \alpha_l, \alpha_r) \not\rightarrow (0, 0, \alpha_l, \alpha_r)$, where $(\alpha_l, \alpha_r) \neq (0, 0)$.

Kühn's attacks use a round structure equivalent to the original one, which is illustrated in Fig. 4; let $[KI_{6j}]_{15-9}$ denote the bits from 9 to 15 of KI_{6j} , $[KI_{6k}]_{8-0}$ denote the bits from 0 to 8 of KI_{6k} , and $KI'_{6j} = [KI_{6j}]_{15-9} || 00 || [KI_{6j}]_{15-9}$, the equivalent subkeys are as follows.

$$\begin{aligned} \text{AKO}_{6k} &= KO_{6k}, \quad k = 1, 2. \\ \text{AKO}_{63} &= KO_{62} \oplus KO_{63} \oplus KI'_{61}. \\ \text{AKO}_{64} &= KO_{62} \oplus KO_{64} \oplus KI'_{61} \oplus KI'_{62}. \end{aligned}$$

$$\begin{aligned} \text{AKO}_{65} &= \text{KO}_{62} \oplus \text{KI}'_{61} \oplus \text{KI}'_{62} \oplus \text{KI}'_{63}. \\ \text{AKI}_{6k} &= [\text{KI}_{6k}]_{8-0}, \quad k = 1, 2, 3. \end{aligned}$$

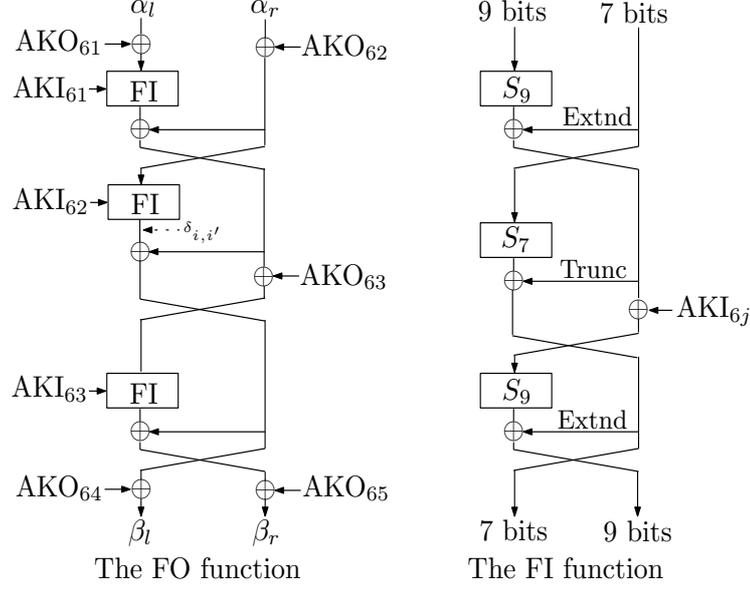


Fig. 4. Impossible differential attack on 6-round MISTY1

MISTY1 has a modified Feistel structure, which is rather different from the “regular” one. Nevertheless, the MISTY1 round structure also allows us to use the early abort technique. As a result, we can improve the first attack due to Kühn, as follows.

1. Choose 2^7 structures of 2^{32} plaintexts $P_i = (x, y, a_i, b_i)$ each, where x and y are 16-bit fixed constants, and a_i and b_i take all the possible 2^{16} values. Keep only the pairs $(P_i, P_{i'})$ with an output difference $(?, ?, \alpha_l, \alpha_r)$. The expected number of the remaining ciphertext pairs is $2^7 \cdot \frac{2^{32 \times 2}}{2} \cdot 2^{-32} = 2^{38}$. (This step is exactly the same as that in Kühn’s attack.)
2. Guess the 41 subkey bits ($\text{AKO}_{61}, \text{AKI}_{61}, \text{AKO}_{62}$) in Round 6. For every remaining ciphertext pair $(C_i, C_{i'})$, the 32-bit difference in the left side is known, say (β_l, β_r) , (β_l and β_r are 16-bit long), so we can compute the difference just after the second FI in the FO by using $(\text{AKO}_{61}, \text{AKI}_{61})$; we denote the difference by $\delta_{i,i'}$, (see Fig. 4). As a consequence, using $\delta_{i,i'}$ we can compute the difference just after the S_7 S-box in the second FI by using AKO_{62} . On the other hand, we know the two inputs to this S_7 S-box for the pair, whose difference is the right 7 bits of α_r . Finally, keep the pair if the

- inputs to the S_7 produce the output difference obtained earlier. This imposes a 7-bit filtering condition; thus about $2^{38} \cdot 2^{-7} = 2^{31}$ pairs are expected to remain for every subkey guess. This step has a time complexity of about $2 \cdot 2^{38} \cdot 2^{41} \cdot \frac{1}{6} \cdot \frac{2}{3} \approx 2^{77}$ 6-round MISTY1 computations.
3. Guess the 9 subkey bits AKI₆₂. For a remaining pair $(C_i, C_{i'})$, with $\delta_{i,i'}$ we can compute the output difference of the second S_9 S-box in the second FI. Keep the pairs which produce these output differences. The expected number of the remaining pairs is $2^{31} \cdot 2^{-9} = 2^{22}$. This step has a time complexity of about $2 \cdot 2^{31} \cdot 2^{50} \cdot \frac{1}{6} \cdot \frac{1}{3} \approx 2^{78}$ 6-round MISTY1 computations.
 4. Guess the 16 subkey bits AKO₆₃. For a remaining pair, with (β_l, β_r) we can compute the difference just after the S_7 S-box of the third FI by using AKO₆₃. Keep the pairs which produce these output differences. The expected number of the remaining pairs is $2^{22} \cdot 2^{-7} = 2^{15}$. This step has a time complexity of about $2 \cdot 2^{22} \cdot 2^{66} \cdot \frac{1}{6} \cdot \frac{1}{3} \approx 2^{85}$ 6-round MISTY1 computations.
 5. Guess the 9 subkey bits AKI₆₃, and check whether or not there is a pair such that the difference just after the third FI is $\beta_l \oplus \beta_r$. If there is such a pair, the guess for $(AKO_{61}, AKI_{61}, AKO_{62}, AKI_{62}, AKO_{63}, AKI_{63})$ is impossible, discard it, and guess another. The expected number of the remaining guesses for the 75 subkey bits is $2^{75} \cdot (1 - 2^{-9})^{2^{15}} \approx 2^{-17}$; thus we can assume it is the correct one. This step has a time complexity of about $2 \cdot 2^{75} \cdot [1 + (1 - 2^{-9}) + \dots + (1 - 2^{-9})^{2^{15}}] \cdot \frac{1}{6} \cdot \frac{1}{3} \approx 2^{81}$ 6-round MISTY1 computations.

Therefore, this attack has a total time complexity of about 2^{85} 6-round MISTY1 computations, significantly lower than the complexity of 2^{106} for Kühn's attack.

6 Summary

In this paper, we observe that, when conducting an impossible differential cryptanalysis on Camellia and MISTY1, their round structures allow us to partially determine whether a candidate pair is right by guessing only a small fraction of the unknown required subkey bits of a relevant round at a time, instead of all of them. This can reduce the computation complexity of an attack, and may allow us to break more rounds. In light of the early abort technique, we improve the previous impossible differential attacks on 12-round Camellia-192 without the FL functions and 6-round MISTY1 without the FL functions, and present impossible differential cryptanalysis of 11-round Camellia-128 without the FL functions and 13-round Camellia-256 without the FL functions, obtaining the best published cryptanalytic results on Camellia and MISTY1.

References

1. Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita, *Camellia: a 128-bit block cipher suitable for multiple platforms — design and analysis*, Proceedings of SAC '00, Lecture Notes in Computer Science 2012, pp. 39–56, Springer-Verlag, 2001.

2. Steve Babbage and Laurent Frisch, On MISTY1 higher order differential cryptanalysis, Proceedings of ICISC '00, Lecture Notes in Computer Science 2015, pp. 22–36, Springer-Verlag, 2001.
3. Eli Biham, Alex Biryukov, and Adi Shamir, Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials, Advances in Cryptology — EUROCRYPT '99, Lecture Notes in Computer Science 1592, pp. 12–23, Springer-Verlag, 1999.
4. Eli Biham, Alex Biryukov, and Adi Shamir, Miss in the middle attacks on IDEA and Khufu, Proceedings of FSE '99, Lecture Notes in Computer Science 1636, pp. 124–138, Springer-Verlag, 1999.
5. Eli Biham and Adi Shamir, Differential cryptanalysis of DES-like cryptosystems, Advances in Cryptology — CRYPTO '90, Lecture Notes in Computer Science 537, pp. 2–21, Springer-Verlag, 1990.
6. CRYPTREC — Cryptography Research and Evaluation Committees, report 2002, <http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html>
7. Yasuo Hatano, Hiroki Sekine, and Toshinobu Kaneko, Higher order differential attack of Camellia(II), Proceedings of SAC '02, Lecture Notes in Computer Science 2595, pp.39–56, Springer-Verlag, 2003.
8. Yeping He and Sihang Qing, Square attack on reduced Camellia cipher, Proceedings of ICICS '01, Lecture Notes in Computer Science 2229, pp. 238–245, Springer-Verlag, 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. Ulrich Kühn, Cryptanalysis of reduced-round MISTY, Advances in Cryptology — EUROCRYPT '01, Lecture Notes in Computer Science 2045, pp. 325–339, Springer-Verlag, 2001.
11. Ulrich Kühn, Improved cryptanalysis of MISTY1, Proceedings of FSE '02, Lecture Notes in Computer Science 2365, pp. 61–75, Springer-Verlag, 2002.
12. Lars K. Knudsen, Truncated and higher order differentials, Proceedings of FSE '94, Lecture Notes in Computer Science 1008, pp. 196–211, Springer-Verlag, 1995.
13. Lars K. Knudsen, DEAL — a 128-bit block cipher, Technical report, Department of Informatics, University of Bergen, Norway, 1998.
14. Lars K. Knudsen and David Wagner, Integral cryptanalysis, Proceedings of FSE '02, Lecture Notes in Computer Science 2365, pp. 112–127, Springer-Verlag, 2002.
15. Seonhee Lee, Seokhie Hong, Sangjin Lee, Jongin Lim, and Seonhee Yoon, Truncated differential cryptanalysis of Camellia, Proceedings of ICISC '01, Lecture Notes in Computer Science 2288, pp. 32–38, Springer-Verlag, 2002.
16. Duo Lei, Li Chao, and Keqin Feng, New observation on Camellia, Proceedings of SAC '05, Lecture Notes in Computer Science 3897, pp. 51–64, Springer-Verlag, 2006.
17. Mitsuru Matsui, Linear cryptanalysis method for DES cipher, Advances in Cryptology — EUROCRYPT '93, Lecture Notes in Computer Science 765, pp. 386–397, Springer-Verlag, 1994.
18. Mitsuru Matsui, New block encryption algorithm MISTY, Proceedings of FSE '97, Lecture Notes in Computer Science 1267, pp. 54–68, Springer-Verlag, 1997.
19. NESSIE — New European Schemes for Signatures, Integrity, and Encryption, final report, <https://www.cosic.esat.kuleuven.be/nessie/Bookv015.pdf>
20. NIST — National Institute of Standards and Technology, Advanced Encryption Standard (AES), FIPS-197, 2001, <http://www.nist.gov/aes>
21. Taizo Shirai, Differential, linear, boomerang and rectangle cryptanalysis of reduced-Round Camellia, Proceedings of The Third NESSIE Workshop, 2002.

22. Makoto Sugita, Kazukuni Kobara, and Hideki Imai, Security of reduced version of the block cipher Camellia against truncated and impossible differential cryptanalysis, *Advances in Cryptology — ASIACRYPT '01, Lecture Notes in Computer Science* 2248, pp. 193–207, Springer-Verlag, 2001.
23. Hidema Tanaka, Kazuyuki Hisamatsu, and Toshinobu Kaneko, Strength of MISTY1 without FL function for higher order differential attack, *Proceedings of AAECC-13, Lecture Notes in Computer Science* 1719, pp. 221–230, Springer-Verlag, 1999.
24. Wenling Wu, Dengguo Feng, and Hua Chen, Collision attack and pseudorandomness of reduced-round Camellia, *Proceedings of SAC '04, Lecture Notes in Computer Science* 3357, pp. 256–270, Springer-Verlag, 2005.
25. Wenling Wu, Wentao Zhang, and Dengguo Feng, Impossible differential cryptanalysis of reduced-round ARIA and Camellia, *Journal of Computer Science and Technology*, Vol. 22(3), pp. 449–456, Springer, 2007. A preliminary version appears as the *Cryptology ePrint Archive*, Report 2006/350.
26. Yongjin Yeom, Sangwoo Park, and Iljun Kim, On the security of Camellia against the square attack, *Proceedings of FSE '02, Lecture Notes in Computer Science* 2356, pp. 89–99, Springer-Verlag, 2002.
27. Yongjin Yeom, Sangwoo Park, and Iljun Kim, A study of integral type cryptanalysis on Camellia, *Proceedings of The 2003 Symposium on Cryptography and Information Security*, pp. 453–456, 2003.