Related-Key Boomerang Attacks on KATAN32/48/64

Takanori Isobe[1], Yu Sasaki[2], and Jiageng Chen[3]

[1] Kobe University
Takanori.Isobe@jp.sony.com
[2] NTT Secure Platform Laboratories
sasaki.yu@lab.ntt.co.jp
[3] Japan Advanced Institute of Science Technology
jg-chen@jaist.ac.jp

Abstract. KATAN/KTANTAN is a family of hardware oriented block ciphers proposed at CHES 2009. Although the KTANTAN family have been broken by a meet-in-the-middle approach, the KATAN family are secure at present. In this paper, we investigate the KATAN family in the related-key boomerang framework with several techniques. By using an efficient differential characteristics search method, long boomerang distinguishers can be built. Furthermore, the key recovery phase is optimized by exploiting several properties of the round function such as the high linearity of the round function and the slow key diffusion. As a result, we can attack 174, 145 and 130 rounds of KATAN32, KATAN48 and KATAN64, which substantially improve the known best results whose attacked rounds are 120, 103, 94 rounds, respectively. Our attacks are confirmed by various experimental verifications, especially, we give concrete right quartets for KATAN32.

Keywords: KATAN32/48/64, ultra lightweight block cipher, related-key attack, boomerang attack, differential.

1 Introduction

KATAN/KTANTAN is a family of lightweight block ciphers designed for extremely resource-constrained devices such as RFID and sensor nodes [1]. After its publication in CHES 2009, the full-round KTANTAN family was theoretically broken by using a meet-in-the-middle approach [2]. The attack takes advantage of the simple key scheduling algorithm for the KTANTAN family. The complexity of the attack was later improved by using the splice-and-cut technique [3]. Armed with related-key model, KTANTAN family can even be broken in practical time [4]. For the KATAN family where the key is loaded into a register and updated in each round, the meet-in-the-middle approach is not likely to work well as the cases of KTANTAN. In the single-key setting, a conditional differential attack is applied to 78, 70 and 68 rounds of KATAN32, KATAN48 and KATAN64, respectively [5]. These results were further improved by using a

C. Boyd and L. Simpson (Eds.): ACISP 2013, LNCS 7959, pp. 268–285, 2013.

variant of the meet-in-the-middle approach to 110, 100 and 94 rounds [6]. Also, a differential-style attack broke the 115-round KATAN32 [7]. Even in the related-key setting, only 120, 103 and 90 rounds for the respective three versions were broken by the conditional differential attack [8]. Given the full 254 rounds, the KATAN family seem to have enough security margin at present. Note that the accelerating key searches for the full KATAN32/48/64 were presented in [9].

In this paper, we further investigate the security of the KATAN family in the related-key boomerang framework. In order to build a long and efficient boomerang distinguisher, we use an efficient differential characteristics search strategy. Generally speaking, this strategy is inspired by observing that there exists 39 consecutive rounds where the related key difference is zero. We call it blank step, and by fixing the starting round of the blank step, we can go backwards and forwards to compute the input and output differences for both E_0 and E_1. Since the key scheduling algorithm is linear for the KATAN family, key difference fixed in E_1 can still be propagated in backwards deterministically. Although a similar strategy was used for conditional differential attacks [5,8], we optimize it for boomerang-type attacks. In particular, we carefully choose sets of input differences which are likely to produce differential characteristics with very high probability, and then exhaustively search for differential characteristics of each input set. The probability for E_0 can be further controlled by adding conditions in the plaintexts. By taking multiple output differences for E_0 and multiple input differences for E_1 into consideration, we are able to build 140, 119 and 113 rounds related-key boomerang distinguisher for the corresponding three versions. Based on the boomerang distinguisher, we further optimize the key recovery phase by exploiting the property of the round function in order to reduce the complexity as well as increasing the number of attacked rounds. The comparison of the attacks against the KATAN family is summarized in Table 1. Our attacks substantially improve previous attacks for all variants, and are confirmed by various experimental verifications, especially, we give the concrete right quartets for KATAN32 which supports the feasibility of the attack.

Outline of the Paper This paper is organized as follows. A description of KATAN and related-key boomerang attack are given in Section 2. The related-key boomerang distinguisher on KATAN32 is shown in Section 3. In Section 4, we present a key recovery attack using the boomerang distinguisher on KATAN32. The analysis of KATAN48/64 is given in Section 5. Finally, we present conclusions in Section 6 with various experimental results showed in Appendix.

2 Preliminaries

2.1 KATAN Block Cipher

The KATAN family [1] is a feedback shift register-based block cipher consisting of three variants : KATAN32, KATAN48, KATAN64, whose block sizes are 32 bits, 48 bits and 64 bits, respectively. All variants use the same LFSR(Linear Feedback Shift Register)-type key scheduling function accepting an 80-bit key.

Table 1. Comparison of attacks against KATAN family

Cipher	Attacking Technique	#Rounds	Time	Data	Mem.	Reference
KATAN32	Differential (SK)	78	2^{76}	2^{16} CP	Not given	[5]
	MitM (SK)	110	2^{77}	138 KP	$2^{75.1}$	[6]
	Differential (SK)	115	2^{79}	138 KP	$2^{75.1}$	[7]
	Differential (RK)	120	2^{31}	Practical (**CP**)	Practical	[8]
	Boomerang (RK)	**172**	$\mathbf{2^{76.2}}$	$\mathbf{2^{27.6}}$ **CP**	$\mathbf{2^{26.6}}$	**Ours**
	Boomerang (RK)	**173**	$\mathbf{2^{77.5}}$	$\mathbf{2^{27.6}}$ **CP**	$\mathbf{2^{26.6}}$	**Ours**
	Boomerang (RK)	**174**	$\mathbf{2^{78.8}}$	$\mathbf{2^{27.6}}$ **CP**	$\mathbf{2^{26.6}}$	**Ours**
KATAN48	Differential (SK)	70	2^{78}	2^{31} CP	Not given	[5]
	MitM (SK)	100	2^{78}	128 KP	2^{78}	[6]
	Differential (RK)	103	2^{25}	Practical (**CP**)	Practical	[8]
	Boomerang (RK)	**145**	$\mathbf{2^{78.5}}$	$\mathbf{2^{38.4}}$ **CP**	$\mathbf{2^{37.4}}$	**Ours**
KATAN64	Differential (SK)	68	2^{78}	2^{32} CP	Not given	[5]
	MitM (SK)	94	$2^{77.68}$	116 KP	$2^{77.68}$	[6]
	Differential (RK)	90	2^{27}	Practical (**CP**)	Practical	[8]
	Boomerang (RK)	**130**	$\mathbf{2^{78.1}}$	$\mathbf{2^{53.1}}$ **CP**	$\mathbf{2^{52.1}}$	**Ours**

SK: Single Key, RK: Related Key, KP: Know Plaintext, CP:Chosen Plaintext.

The key scheduling function expands an 80-bit user-provided key k_i $(0 \leq i < 80)$ into a 508-bit subkey sk_i $(0 \leq i < 508)$ by the following linear operations,

$$sk_i = \begin{cases} k_i \ (0 \leq i < 80), \\ k_{i-80} \oplus k_{i-61} \oplus k_{i-50} \oplus k_{i-13} \ (80 \leq i < 508). \end{cases}$$

These operations are expressed as an 80-bit LFSR whose polynomial is $x_{80} + x_{61} + x_{50} + x_{13} + 1$ as shown in Fig 1.

In the round function, each bit of a plaintext is loaded into registers L_1 and L_2. Then, these are updated as follows:

$$\begin{aligned} f_a(L_1) &= L_1[x_1] \oplus L_1[x_2] \oplus (L_1[x_3] \cdot L_1[x_4]) \oplus (L_1[x_5] \cdot IR) \oplus k_a, \\ f_b(L_2) &= L_2[y_1] \oplus L_2[y_2] \oplus (L_2[y_3] \cdot L_2[y_4]) \oplus (L_2[y_5] \cdot L_2[y_6]) \oplus k_b, \\ L_1[i] &= L_1[i-1] \ (1 \leq i < |L_1|), \quad L_1[0] = f_b(L_2), \\ L_2[i] &= L_2[i-1] \ (1 \leq i < |L_2|), \quad L_2[0] = f_a(L_1), \end{aligned}$$

where $\oplus$ and $\cdot$ are bitwise XOR and AND operations, respectively, and $L[x]$ denotes the x-th bit of L, IR is the round constant value defined in the specification, and k_a and k_b are two subkey bits. Table 2 shows the detailed parameters of KATAN32/48/64. For round i, k_a and k_b correspond to $sk_{2(i-1)}$ and $sk_{2(i-1)+1}$, respectively. After 254 rounds (from 1 to 254) values of registers are output as a ciphertext. Fig. 2 illustrates the round function of KATAN32.

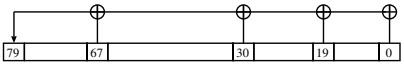

Fig. 1. Key scheduling function of KATAN32/48/64

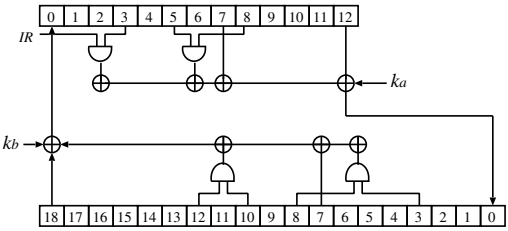

Fig. 2. Round function of KATAN32

Table 2. Parameters of KATAN family

Algorithm	$\|L_1\|$	$\|L_2\|$	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4	y_5	y_6
KATAN32	13	19	12	7	8	5	3	18	7	12	10	8	3
KATAN48	19	29	18	12	15	7	6	28	19	21	13	15	6
KATAN64	25	39	24	15	20	11	9	38	25	33	21	14	9

2.2 Related-Key Boomerang Attack

The related-key boomerang attack [10,11,12] is a combination of the boomerang attack [13], and the related-key differential attack [14,15,16].

Boomerang-Type Attack. The main idea behind the boomerang attack [13] is to use two short differentials with high probability instead of one long differential with low probability. Suppose that a block cipher with n-bit block and k-bit key, $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$, is expressed as a cascade cipher $E = E_1 \circ E_0$, where E_0 has a differential $\alpha \to \beta$ with probability p, and E_1 has a differential $\gamma \to \delta$ with probability q. Then, the distinguisher is mounted as follows:

1 : Ask for the ciphertexts $C_1 = E(P_1)$ and $C_2 = E(P_2)$, where $P_2 = P_1 \oplus \alpha$.
2 : Ask for the plaintexts $P_3 = E^{-1}(C_3)$ and $P_4 = E^{-1}(C_4)$, where $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$.
3 : Check whether $P_3 \oplus P_4 = \alpha$.

Here, E satisfies the condition of $P_3 \oplus P_4 = \alpha$ with probability of p^2q^2, while that of a random permutation is 2^{-n}. Note that the attack can be mounted for all possible β's and γ's simultaneously. Therefore, the probability is improved to $\hat{p}^2\hat{q}^2$ from p^2q^2, where $\hat{p} = \sqrt{\sum_\beta Pr^2[\alpha \to \beta]}$ and $\hat{q} = \sqrt{\sum_\gamma Pr^2[\gamma \to \delta]}$.

The amplified boomerang attack converts the adaptive setting into the non-adaptive one [17]. It exploits the birthday paradox in the middle round. An attacker encrypts many plaintext pairs with a difference α, and collects plaintext/ciphertext quartets. Then, she searches for right quartets in the form of $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$. For E, this event occurs if the following three conditions are satisfied:

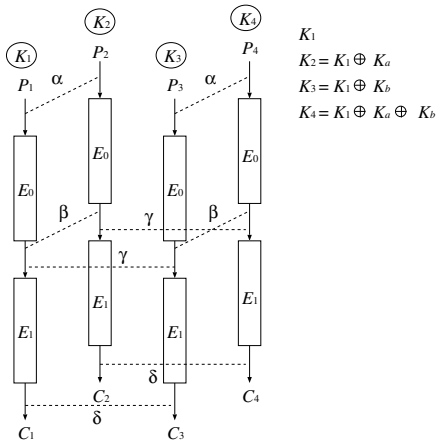

Fig. 3. Related-key boomerang quartet

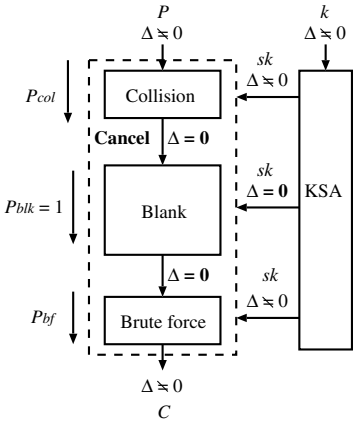

Fig. 4. Strategy for finding differential characteristics

Condition 1 : $E_0(P_1) \oplus E_0(P_2) = E_0(P_3) \oplus E_0(P_4) = \beta$,
Condition 2 : $E_0(P_1) \oplus E_0(P_3)$(or $E_0(P_2) \oplus E_0(P_4)) = \gamma$,
Condition 3 : $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$.

The probability that a quartet is the right one is $2^{-n}p^2q^2$. For a random permutation, this event occurs with probability of 2^{-2n}. Thus, if $pq > 2^{-n/2}$, we can distinguish E from a random permutation. Given N plaintext pairs having α difference, there are $\binom{N}{2} \times 2 \approx N^2$ quartets. Thus, the expected number of right quartets in N pairs is $N^2 \cdot 2^{-n}p^2q^2$. The rectangle attack [18] exploits all β and γ to improve the amplified boomerang attack. If $\hat{p}^2\hat{q}^2 > 2^{-n/2}$, this distinguisher works. Though the rectangle attack requires a large amount of data, it can perform a key recovery phase in the non-adaptive setting.

In this paper, we refer boomerang-type attack using amplified and rectangle techniques to boomerang attack for sake of simplicity.

Related-Key Boomerang Attack. The related-key boomerang attack [10,11,12] additionally uses key differences. See Fig. 3 for its illustration. Assume that E_0 has a differential $\alpha \to \beta$ under a key difference ΔK_a with probability $\hat{p}$, and E_1 has a differential $\gamma \to \delta$ under a key difference ΔK_b with probability $\hat{q}$. A related-key distinguisher is constructed by using four different unknown keys, K_1, $K_2 = K_1 \oplus K_a$, $K_3 = K_1 \oplus K_b$, $K_4 = K_1 \oplus K_a \oplus K_b$, as follows:

1 : Ask N ciphertext pairs (C_1, C_2), where $C_1 = E_{K_1}(P_1)$, $C_2 = E_{K_2}(P_2)$ and $P_1 \oplus P_2 = \alpha$. Define the set of these pairs as S.
2 : Ask N ciphertext pairs (C_3, C_4), where $C_3 = E_{K_3}(P_3)$, $C_4 = E_{K_4}(P_4)$ and $P_3 \oplus P_4 = \alpha$. Define the set of these pairs as T.
3 : Find right quartets satisfying the following conditions from S and T:
$P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ and $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$,

Table 3. Output differences of z for each input value (x, y) and its difference

Value of	Difference of (x, y)		
(x, y)	(0,1)	(1,0)	(1,1)
(0,0)	0	0	1
(0,1)	0	1	0
(1,0)	1	0	0
(1,1)	1	1	1

Table 4. Sets of input diff. with Hamming weight 2

set	key difference	no-difference subkeys	plaintext differences	P_{col}
0	0, 19	$20 - 98$	$L_2[9], L_1[12]$	2^{-2}
1	1, 20	$21 - 99$	$L_2[18], L_1[2, 7, 12]$	2^{-3}
2	2, 21	$22 - 100$	$L_2[8], L_1[11]$	2^{-3}
3	3, 22	$23 - 101$	$L_2[17], L_1[1, 6, 11]$	2^{-3}
4	4, 23	$24 - 102$	$L_2[7, 18], L_1[10]$	2^{-3}
5	5, 24	$25 - 103$	$L_2[16], L_1[0, 5, 10]$	2^{-4}
6	6, 25	$26 - 104$	$L_2[6, 17], L_1[9]$	2^{-3}
7	7, 26	$27 - 105$	$L_2[15, 18], L_1[4, 9]$	2^{-3}
8	8, 27	$28 - 106$	$L_2[5, 16], L_1[8]$	2^{-4}
9	9, 28	$29 - 107$	$L_2[14, 17], L_1[3, 8]$	2^{-5}
10	10, 29	$30 - 108$	$L_2[4, 15], L_1[7, 12]$	2^{-4}

3 Related-Key Boomerang Distinguisher on KATAN32

In this section, we introduce an effective search strategy for finding good related-key differential characteristics. This technique exploits the linearity of the key scheduling and the low dependency of subkey bits. Although a similar search strategy was used in [5,8], we optimize it for a boomerang-type attack.

3.1 Differential Properties of KATAN

Round Function. Let us consider an XOR differential property of the round function of KATAN in which there are four nonlinear components, *i.e.*, AND operations. Table 3 shows the differential property of the AND operation whose inputs are x, y and the output is z, namely $z = x \cdot y$. For example, for the value $(1, 0)$ and the difference $(1, 0)$, the difference of z is obtained as $(x \cdot y) \oplus ((x \oplus 1) \cdot (y \oplus 0)) = (1 \cdot 0) \oplus (0 \cdot 1) = 0$. From Table 3, when input values have any differences, the output also has a difference with probability $2^{-1} (= 6/12)$.

Besides, one AND operation takes IR as one of the input bits. If one of input bits is public and constant, the corresponding output difference is deterministic. Thus, we can focus on only *three* AND operations as nonlinear operations.

Key Scheduling Function. The key scheduling function employs only linear operations based on the LFSR. Then, we obtain the following observation.

> **Observation.** *Choosing input key differences properly, 79 consecutive subkey bits have no differences after the key scheduling function.*

Since the key scheduling function is the 80-bit LFSR-type construction, any 80 consecutive subkeys surely contain some differences if the key has differences. However, if only one bit of k_i $(0 \leq i \leq 18)$ has a difference, there is no differences in $k_{i+1} - k_{i+79}$, because k_i is not used until k_{i+80}. As for the other case, Table 4 shows all possible sets of 2-bit input key differences producing such 79-bit no differences subkeys. For example, assuming that k_0 and k_{19} have differences (set 0), $k_{20} - k_{98}$ do not have differences because the difference k_0 is canceled by k_{19} when it is used for computing k_{80}. The same event occurs in other sets 1-10.

Note that, for all 79 consecutive subkey bits, we can generate the subkey difference which does not make any difference for the target 79 subkey bits. This can be done by the kernel computing approach in [19]. However these sets do not give advantage compared to the sets 1-10, and thus we omit the details.

3.2 Strategy for Finding Differential Characteristics

We introduce an effective search strategy for finding good related-key differential characteristics. It is well-suited for boomerang-type attacks in terms of short differential characteristics with very high probability. In general, it is difficult to find good differential characteristics for a bit-oriented cipher due to the large search space. Besides, the related-key setting where key differences are additionally inserted makes it more difficult. In order to get rid of this problem, our strategy is focusing on particular input differential sets which are expected to give good characteristics for boomerang-type attacks.

The differential characteristic search strategy consists of a collision step, a blank step and a brute force step as shown in Fig.4.

Collision Step : Plaintext difference and key difference cancel each other.
Blank Step : No difference exists in registers and inserted subkeys.
Brute Force Step : Subkey differences propagate to the registers.

The key idea of this strategy is to construct the rounds having no difference called *blank round*. Since the blank round does not reduce the differential probability, *i.e.*, differential probability of such rounds is one, we expect to obtain differential characteristics with high probability. For constructing a long blank round, we utilize the observation 1: *we can set 79 consecutive subkey bits having no difference.* If there is no difference in registers where these 79-bit subkeys are used, the blank round can be easily constructed. In other words, we properly choose plaintext difference for canceling out subkey differences just before the blank round. Table 4 shows the plaintext differences for canceling the corresponded input key differences before the blank round and its probability. After the blank round, we search for all differential characteristics. As mentioned before, we regard three AND operations as nonlinear components. Let P_{col}, P_{blk} and P_{bf} be the differential probability of each step, respectively. The whole differential characteristic probability is calculated as $P_{col} \cdot P_{blk} \cdot P_{bf}$, where $P_{blk} = 1$.

Input key differences are restricted to the set satisfying the property of the observation 1. Then, plaintext differences are also determined from the set of input key differences for constructing the blank round (see Table 4).

3.3 Related-Key Boomerang Distinguisher on 140-Round KATAN32

Using the efficient differential characteristics search, we obtain the maximum probability of differential characteristics of each input set in E_0 starting from round 1 (see Table 5).

Table 5. Maximum probability of differential characteristics of each set in E_0

Round	**set0**	**set1**	**set2**	**set3**	**set4**	**set5**	**set6**	**set7**	**set8**	**set9**	**set10**
65	2^{-9}	2^{-9}	2^{-7}	2^{-8}	2^{-7}	2^{-8}	2^{-7}	2^{-7}	2^{-7}	2^{-7}	2^{-7}
66	2^{-10}	2^{-10}	2^{-7}	2^{-9}	2^{-7}	2^{-9}	2^{-7}	2^{-8}	2^{-8}	2^{-8}	2^{-8}
67	2^{-12}	2^{-10}	2^{-8}	2^{-10}	2^{-7}	2^{-10}	2^{-7}	2^{-9}	2^{-8}	2^{-9}	2^{-9}
68	2^{-13}	2^{-11}	2^{-9}	2^{-10}	2^{-8}	2^{-11}	2^{-7}	2^{-11}	2^{-8}	2^{-10}	2^{-10}
69	2^{-14}	2^{-12}	2^{-10}	2^{-12}	2^{-9}	2^{-11}	2^{-8}	2^{-12}	2^{-8}	2^{-11}	2^{-11}
70	2^{-15}	2^{-12}	2^{-12}	2^{-12}	2^{-10}	2^{-12}	2^{-9}	2^{-12}	2^{-9}	2^{-12}	2^{-12}
71	2^{-16}	2^{-13}	2^{-13}	2^{-12}	2^{-12}	2^{-13}	2^{-10}	2^{-14}	2^{-10}	2^{-12}	2^{-12}

Table 6. Maximum probability of differential characteristics of each set in E_1

Round	**set0**	**set1**	**set2**	**set3**	**set4**	**set5**	**set6**	**set7**	**set8**	**set9**	**set10**
65	2^{-6}	2^{-10}	2^{-7}	2^{-8}	2^{-9}	2^{-8}	2^{-7}	2^{-7}	2^{-8}	2^{-7}	2^{-6}
66	2^{-7}	2^{-11}	2^{-7}	2^{-9}	2^{-9}	2^{-9}	2^{-7}	2^{-8}	2^{-9}	2^{-8}	2^{-7}
67	2^{-8}	2^{-11}	2^{-8}	2^{-10}	2^{-10}	2^{-11}	2^{-7}	2^{-9}	2^{-10}	2^{-9}	2^{-8}
68	2^{-9}	2^{-13}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-7}	2^{-10}	2^{-10}	2^{-10}	2^{-8}
69	2^{-11}	2^{-13}	2^{-10}	2^{-12}	2^{-13}	2^{-12}	2^{-8}	2^{-11}	2^{-11}	2^{-11}	2^{-8}
70	2^{-12}	2^{-13}	2^{-12}	2^{-12}	2^{-14}	2^{-13}	2^{-9}	2^{-11}	2^{-12}	2^{-12}	2^{-8}
71	2^{-15}	2^{-14}	2^{-13}	2^{-12}	2^{-15}	2^{-14}	2^{-10}	2^{-12}	2^{-14}	2^{-12}	2^{-9}

To construct a distinguisher, we choose 70 rounds of set 8 whose probability is highest of all the sets. E_0 has 8 characteristics with probability $p = 2^{-9}$, 16 characteristics with probability $p = 2^{-10}$, 16 characteristics with probability $p = 2^{-11}$ and 64 characteristics with probability $p = 2^{-12}$, which are generated from the same input. Thus, the overall probability for E_0 is

$$\hat{p} = \sqrt{(2^{-9})^2 \cdot 8 + (2^{-10})^2 \cdot 16 + (2^{-11})^2 \cdot 16 + (2^{-12})^2 \cdot 64} \approx 2^{-7.1}.$$

Table 11 in Appendix gives a single differential trail of E_0 with probability of 2^{-9}, where round 0 means initial differences, *i.e.*, differences of a plaintext.

Since KATAN employs the LFSR-based key scheduling, all 508 subkey bits can be calculated from any consecutive 80 subkey bits. It means that we can use the efficient differential characteristics search strategy from any round by regarding the consecutive 80 subkey bits as the master key bits. Thus, we search for differential characteristics of E_1 from round 71 with the same strategy.

Table 6 shows the maximum probability of differential characteristics of each set in E_1 starting from round 71. We choose set 1 as E_1 with probability 2^{-8}. In addition, E_1 has 4 characteristics with probability 2^{-8}, 8 characteristics with probability 2^{-9} and 32 characteristics with probability 2^{-10}, which produce the same output difference. Thus, the total probability for E_1 is estimated as

$$\hat{q} = \sqrt{(2^{-8})^2 \cdot 4 + (2^{-9})^2 \cdot 8 + (2^{-10})^2 \cdot 32} \approx 2^{-6.5}.$$

Table 12 in Appendix gives a single differential trail of E_1 with probability 2^{-8}.

Combining these two-type differential characteristics, 140 (=70+70)-round related-key boomerang distinguisher can be constructed with probability of

$$\hat{p}^2 \cdot \hat{q}^2 = (2^{-7.1})^2 \cdot (2^{-6.5})^2 = 2^{-27.2} \ (> 2^{-32}).$$

The probability of the boomerang distinguisher, $2^{-27.2}$, is possible to verify practically. We performed the experiment on a standard PC and found right quartets within a few minutes. One example is shown in Table 10 in Appendix.

4 Related-Key Recovery Attack on KATAN32

In this section, a related-key attack on KATAN32 is proposed given the 140-round boomerang distinguisher. One of the challenging problems is how to reduce the candidate quartets. This is usually achieved by studying the propagation of the difference to the ciphertext in order to filter out definitely wrong quartets. For the KATAN family, this may not be the best option. When we extend the attacking rounds as long as possible, the difference propagation will leave us with no clue. Instead, we try to choose plaintext so that the characteristic for the first several rounds are always satisfied. This strategy is also used in previous KATAN attacks [5,8]. We further optimize the key recovery phase by exploiting the property of the round function, in order to reduce the complexity as well as increasing the number of attacked rounds.

4.1 Conditions for Chosen Plaintexts

In the collision step for E_0, we have calculated that $p_{col} = 2^{-4}$. Recall that for two inputs to an AND gate, one input with value 1 will guarantee the propagation of the difference from the other input, and difference will disappear when the value is fixed to 0. Thus we can assure the difference propagation with probability 1 by fixing some of the plaintext bits. For KATAN32, the probability for the collision steps can be increased to 1. The conditions on plaintext bits are $L_2[0] = L_2[3] = L_2[7] = L_1[5] = 0$, and the increased probability for E_0 is

$$\hat{p} = \sqrt{(2^{-9+4})^2 \cdot 8 + (2^{-10+4})^2 \cdot 16 + (2^{-11+4})^2 \cdot 16 + (2^{-12+4})^2 \cdot 64} \approx 2^{-3.1}.$$

This indicates that we can expect one right quartet in $2^{51.2} (= (2^{-3.1})^2 \cdot (2^{-6.5})^2 \cdot 2^{-32})$. As a result, the number of quartet candidates is reduced to $2^{51.2}$.

4.2 Optimizing Key Recovery Phase

Suppose that we append x rounds to the end of the 140-round distinguisher. The attacker queries N pairs of plaintexts to oracles with K_1 and K_2. She also queries N pairs of plaintexts to oracles with K_3 and K_4. Then, N^2 quartets are constructed. We set $N \leftarrow \hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{n/2}$ so that a right quartet is generated.

To recover subkeys for the last x rounds with a straight-forward method, the attacker guesses all subkeys for the last x rounds, and performs partial decryptions until the end of the 140-round distinguisher for each of N^2 quartets. Let g_i, where $i \in \{1, 2, 3, 4\}$ be a set of subkey bits used in the last x

rounds for the K_i oracle. Because KATAN uses two subkey bits in each round, each g_i contains $2x$ subkey bits. We denote the x-round partial decryption for a ciphertext C_i with a guessed key g_i by $D_{g_i}(C_i)$. Note that if the guess for K_1 oracle, g_1, is determined, the corresponding g_2, g_3, and g_4 are determined uniquely. If the guessed value is correct, the attacker will find one quartet such that $D_{g_1}(C_1) \oplus D_{g_3}(C_3) = D_{g_2}(C_2) \oplus D_{g_4}(C_4) = \delta$. If such a quartet is not found, the guess is wrong. Unfortunately, the complexity of this approach is too high. Let $\#g$ be the number of subkey bits in each of g_i, namely $2x$. The approach requires $N^2 \cdot \#g \cdot 4$ partial decryptions, where a factor of N^2 is too high.

Pairwise Approach. We propose a more efficient method. For each guess of g_1 and corresponding g_2, g_3, g_4, we perform the partial decryption for N pairs of (C_1, C_2) and N pairs of (C_3, C_4) independently, and identify the right quartet by checking their match as follows:

1. Make a guess for g_1 and determine the corresponding values for g_2, g_3, g_4.
2. For all N pairs of (C_1, C_2), compute $(D_{g_1}(C_1) \oplus \delta, D_{g_2}(C_2) \oplus \delta)$ and store them in a table with N entries.
3. For all N pairs of (C_3, C_4), compute $(D_{g_3}(C_3), D_{g_4}(C_4))$ and store them in another table.
4. If the guess is correct, a match is found. Otherwise, the guess is discarded.

This method requires only $N \cdot \#g \cdot 2$ partial decryptions for Step 2 and Step 3 respectively, in total $N \cdot \#g \cdot 4$ partial decryptions. The memory requirement is $2N$ state. The memory for Step 3 can be saved by checking the match as soon as we obtain a pair. Each guess is judged as a right-key candidate if one of N^2 quartets satisfies two n-bit relations δ. We denote this probability by P_{right}, which is $N^2 \cdot 2^{-2n}$. After the analysis, the key space will be $\#g \cdot P_{right} = \#g \cdot N^2 \cdot 2^{-2n}$. The remaining key space will be later examined by the exhaustive search.

Exploiting Linear Subkey Insertion. We further optimize the attack by exploiting the round function structure. Recall Fig. 2. If the output *value* for some round r is known, the input *difference* for round r can be computed without guessing subkeys. This is because the 1-round decryption uses subkey values only in the linear operation. The situation continues until unknown values are used as an input of AND operations. In the end, the difference after the x-round decryption can be computed only with guessing subkeys for the last $x-4$ rounds.

Partial Matching. Another optimization is possible by exploiting the property that only 2 bits are updated in each decryption round. Let us see what will happen if we go back 5 rounds without guessing subkeys. As mentioned above, the difference in all bits can be computed up to 4 rounds. In the next round, the attacker cannot compute the difference of the updated bit $L_1[12]$, while she knows the difference of the other 31 bits ($L_2[18]$ can be computed at this stage). Hence, the match can be performed for 31 bits. The analysis is summarized in

Table 7. Partial-matching technique for KATAN32

#Skipped rounds	Number of bits with unknown differences			P_{right}
	L_1	L_2	Total	
1–4	0	0	0	$N^2 \cdot 2^{-64}$
5	1	0	1	$N^2 \cdot 2^{-62}$
6	2	0	2	$N^2 \cdot 2^{-60}$
7	3	1	4	$N^2 \cdot 2^{-56}$
$r(\geq 6)$	$r-4$	$r-6$	$2r-10$	$N^2 \cdot 2^{-84+4r}$

If one subkey bit for the first skipped round is guessed, P_{right} decreases by 2^2.

Table 7. Let r be the number of rounds which we compute without guessing subkeys. Let z be the number of bits with unknown difference. The match is performed for $32 - z$ bits. From Table 7, $z = 2r - 10$ for $r \geq 6$. Because 2 pairs exist in a quartet, P_{right} is $N^2 \cdot 2^{-2(32-z)}$, which is $N^2 \cdot 2^{-84+4r}$. As long as P_{right} is small enough, subkeys can be recovered faster than the exhaustive search.

The idea of checking the difference only for a part of the state is similar to the early abort technique [20]. Our idea is different because the pairwise approach is used and the match of difference cannot be checked round by round.

Partial Key Guessing. The last technique for the optimization is partially guessing a subkey, *i.e.*, only guessing 1 bit of a subkey in the first skipped rounds. In Table 7, this makes the number of unknown bits be $2r - 11$ and P_{right} be $N^2 \cdot 2^{-86+4r}$ when $r \geq 6$. Intuitively, the technique increases the computational complexity by 1 bit due to the additional guessed bit, while it increases the efficiency of the filtering function by 2 bits due to two pairs in a quartet.

4.3 Attack Procedure and Complexity Evaluation

We append $x = 34$ rounds to the end of the 140-round distinguisher. The number of rounds which we do not guess subkey values, r, is 8, but we use the partial key guessing technique. Therefore, $\#g = 53$, where each g_i consists of 52 bits for the last 26 rounds and 1 bit of subkey (either bit is fine) for the 27th last round.

1. Choose $N = 2^{25.6}$ plaintext pairs (P_1, P_2) so that $P_1 \oplus P_2 = \alpha$ and satisfy the 4-bit conditions $L_2[0] = L_2[3] = L_2[7] = L_1[5] = 0$. Query them to the oracles with K_1 and K_2, and store the corresponding $2^{25.6}$ pairs of (C_1, C_2).
2. Do the same for (P_3, P_4) to obtain $N = 2^{25.6}$ ciphertext pairs (C_3, C_4).
3. Guess g_1 and the corresponding g_2, g_3, g_4. For each guess, do as follows.
   (a) For $2^{25.6}$ pairs of (C_1, C_2), decrypt them for 26 rounds. Then, further decrypt them by 8 rounds to obtain differences in $32 - (2 \times 8 - 11) = 27$ bits, and take the XOR with δ. Store them in a table with $2^{25.6}$ entries.
   (b) For $2^{25.6}$ pairs of (C_3, C_4), do as follows.
      i. Similarly decrypt the pair for $26 + 8 = 34$ rounds to obtain the differences in 27 bits.
      ii. Check if the match exists between the stored values. If no match is found, delete the guess from the candidate. Otherwise, do as follows.

iii. For exhaustive guesses of $80 - 53 = 27$-bit subkeys which are not guessed yet, check the correctness of the guess by using any pair of plaintext and ciphertext (32-bit match). It it passes the check, then further check the correctness of the guess with two more plaintext-ciphertext pairs. If it passes all checks, output it as the correct key.

For Step 1 and 2, we need $4 * 2^{25.6} = 2^{27.6}$ chosen plaintexts. Step 3a requires $2^{53+25.6} \cdot 2 \cdot 34/174 \approx 2^{77.25}$ 174-round KATAN32 computations. The memory requirement for Step 3a is about $2 \cdot 2^{25.6} = 2^{26.6}$ state values. Step 3(b)i also requires $2^{77.25}$ computations. After Step 3(b)ii, $2^{53} \cdot P_{right} = 2^{53} \cdot (2^{51.2} \cdot 2^{-86+4 \cdot 8}) = 2^{50.2}$ key candidates will remain. Step 3(b)iii requires $2^{50.2+27} = 2^{77.2}$ 174-round KATAN32 computations for the first plaintext-ciphertext pair. Only $2^{77.2-32} = 2^{45.2}$ key candidates are examined for the second pair, and only $2^{45.2-32} = 2^{13.2}$ candidates are examined for the third pair. Hence, the complexity for Step 3(b)iii is $2^{77.2} + 2^{45.2} + 2^{13.2} \approx 2^{77.2}$ 174-round KATAN32 computations.

In summary, the data complexity is $2^{27.6}$ chosen plaintexts, the time complexity is $2^{77.25} + 2^{77.25} + 2^{77.2} \approx 2^{78.8}$ 174-round KATAN32 computations. The memory requirement is $2^{25.6}$ state.

Note that our attack succeeds only if the right quartet is obtained *i.e.*, the differential with a probability of $2^{-51.2}$ is satisfied with $2^{51.2}$ quartets. Hence, the success probability of our attack is $1 - 1/e \approx 0.63$. On the other hand, the success probability of the brute force attack with $2^{78.8}$ trials is 0.44. Hence, our attack is better than the brute force attack with the same complexity.

Also note that the advantage of our attack becomes clearer if the number of rounds is reduced more. For example, the complexity for 173 or 172 rounds is $2^{77.5}$ or $2^{76.2}$ computations, respectively, with the same data and memory.

5 Related-Key Boomerang Attack on KATAN48/64

5.1 Differential Characteristics and Plaintext Conditions

First we give differential characteristic for KATAN48. Similar to KATAN32, we start from finding collision steps, and by changing the starting point of the collision steps, we go backwards to derive the input differences and key differences. As a result we build a 119-round boomerang distinguisher for KATAN48. Table 13 and 14 in Appendix demonstrate one characteristic for E_0 and E_1. We use a fixed characteristic between rounds 1 to 49 of E_0 and rounds 70 to 119 for E_1, while we use a differential for the other rounds. In total, for E_0 there are 32 characteristics with probability 2^{-14}, 128 characteristics with probability 2^{-15} and 128 characteristics with probability 2^{-16}. For E_1 there are 128 characteristics with probability 2^{-12}. As a result, $\hat{p} = \sqrt{(2^{-14})^2 \cdot 32 + (2^{-15})^2 \cdot 128 + (2^{-16})^2 \cdot 128} = 2^{-10.9}$, $\hat{q} = \sqrt{(2^{-12})^2 \cdot 128} = 2^{-8.5}$.

Differential characteristics for E_0 and E_1 of KATAN64 are summarized in Table 15 and 16 in Appendix. Due to the more scrambling in each round, the number of the collision steps and the brute force steps are reduced. We use a fixed characteristic between rounds 1 to 46 of E_0 and rounds 103 to 113 for

Table 8. Partial-matching for KATAN48

#skipped rounds	#bits with unknown diff. L_1	L_2	Total	P_{right}
1	0	0	0	$N^2 \cdot 2^{-96}$
2	1	0	1	$N^2 \cdot 2^{-94}$
3	3	0	3	$N^2 \cdot 2^{-90}$
4	5	1	6	$N^2 \cdot 2^{-84}$
5	7	3	10	$N^2 \cdot 2^{-76}$
6	9	5	14	$N^2 \cdot 2^{-68}$
$r(\geq 4)$	$2r-3$	$2r-7$	$4r-10$	$N^2 \cdot 2^{-116+8r}$

If one subkey bit for the first skipped round is guessed, P_{right} decreases by 2^4.

Table 9. Partial-matching for KATAN64

#skipped rounds	#bits with unknown diff. L_1	L_2	Total	P_{right}
1	0	0	0	$N^2 \cdot 2^{-128}$
2	2	1	3	$N^2 \cdot 2^{-122}$
3	5	4	9	$N^2 \cdot 2^{-110}$
4	8	7	15	$N^2 \cdot 2^{-98}$
$r(\geq 3)$	$3r-4$	$3r-5$	$6r-9$	$N^2 \cdot 2^{-146+12r}$

If one subkey bit for the first skipped round is guessed, P_{right} decreases by 2^6.

E_1, while we use a differential for the other rounds. For E_0 there are 64, 256, 512, 1024, and 1024 characteristics with probability 2^{-16}, 2^{-17}, 2^{-18}, 2^{-19}, and 2^{-20}, respectively. For E_1 there are 4, 24, 88, 224, 416, 608, 704, 640, and 256 characteristics with probability 2^{-16}, 2^{-17}, 2^{-18}, 2^{-19}, 2^{-20}, 2^{-21}, 2^{-22}, 2^{-23}, and 2^{-24}, respectively. As a result, $\hat{p} = 2^{-12.25}$, and $\hat{q} = 2^{-13.8}$.

The probabilities of the collision steps of E_0 for KATAN48/64 are both 2^{-7}, but this can be improved by 2^7 by choosing the plaintext satisfying the conditions. The conditions are given below along with the improved probability for $\hat{p}$. $\hat{q}$ is not affected by the chosen plaintext.

KATAN48. Conditions: $L_2[0] = L_2[1] = L_2[2] = L_2[11] = L_2[17] = 0, L_2[10] \neq L_2[18]$. $\hat{p} = \sqrt{(2^{-14+7})^2 \cdot 32 + (2^{-15+7})^2 \cdot 128 + (2^{-16+7})^2 \cdot 128} = 2^{-3.9}$. We expect $2^{72.8} (= (2^{3.9})^2 \cdot (2^{8.5})^2 \cdot 2^{48})$ quartets before a right one shows up.

KATAN64. Conditions: $L_2[6] = L_2[7] = L_2[8] = L_2[21] = L_2[30] = 0, L_2[20] \neq L_2[32], L_2[19] \neq L_2[31]$. $\hat{p}$ becomes $2^{-5.25}$. We expect $2^{102.1} (= (2^{5.25})^2 \cdot (2^{13.8})^2 \cdot 2^{64})$ quartets before a right one shows up.

5.2 Optimization and Summary of Key Recovery Attacks

The overall strategy is the same as the one for KATAN32. The only difference from KATAN32 is the impact of the partial-matching technique, which comes from the different register sizes $|L_1|, |L_2|$ and input-bit positions for AND operations. The results are summarized in Table 8 and Table 9.

145-Round KATAN48. The attack generates $\hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{48/2} = 2^{3.9+8.5+24} = 2^{36.4}$ pairs of (P_1, P_2), and $2^{36.4}$ pairs of (P_3, P_4). This makes $2^{72.8}$ quartets, which include a right quartet with probability 0.63. We append 26 rounds after the 119-round distinguisher. Hence, 145 rounds are attacked. In the key recover phase, we guess 42 bits of subkeys for the last 21 rounds. Therefore, the number of skipped steps, r, is 5. This makes the time complexity for the analysis for P_1, P_2 pairs be $2^{36.4+42} \cdot 2 \cdot 26/145 \approx 2^{76.9}$ 145-round KATAN48 computations. The memory

requirement is $2 \cdot 2^{36.4} = 2^{37.4}$ state values. The analysis for P_3, P_4 pairs also requires $2^{76.9}$ 145-round KATAN48 computations. P_{right} is $2^{72.8} \cdot 2^{-76} = 2^{-3.2}$. Hence, the complexity for the exhaustive check becomes $2^{80} \cdot P_{right} = 2^{76.8}$. In the end, the data complexity is $4 \cdot 2^{36.4} = 2^{38.4}$ chosen plaintexts. The computational complexity is $2^{76.9} + 2^{76.9} + 2^{76.8} \approx 2^{78.5}$ 145-round KATAN48 computations. The success probability of our attack is 0.63, while the success probability of the brute force attack with the same complexity is 0.35.

130-Round KATAN64. The attack generates $\hat{p}^{-1} \cdot \hat{q}^{-1} \cdot 2^{64/2} = 2^{5.25+13.8+32} = 2^{51.05}$ pairs of (P_1, P_2), and $2^{51.05}$ pairs of (P_3, P_4). This makes $2^{102.1}$ quartets, which include a right quartet with probability 0.63. We append 17 rounds after the 113-round distinguisher. Hence, 130 rounds are attacked. In the key recover phase, we guess 28 bits of subkeys for the last 14 rounds. Therefore, the number of skipped steps, r, is 3. This makes the time complexity for the analysis for P_1, P_2 pairs be $2^{51.05+28} \cdot 2 \cdot 17/130 \approx 2^{77.1}$ 130-round KATAN64 computations. The memory requirement is $2 \cdot 2^{51.05} \approx 2^{52.1}$ state values. The analysis for P_3, P_4 pairs also requires $2^{77.1}$ 130-round KATAN64 computations. P_{right} is $2^{102.1} \cdot 2^{-110} = 2^{-7.9}$. Hence, the complexity for the exhaustive check becomes $2^{80} \cdot P_{right} = 2^{72.1}$. In the end, the data complexity is $4 \cdot 2^{51.05} \approx 2^{53.1}$ chosen plaintexts. The computational complexity is $2^{77.1} + 2^{77.1} + 2^{72.1} \approx 2^{78.1}$ 130-round KATAN64 computations. The success probability of our attack is 0.63, while the success probability of the brute force attack with the same complexity is 0.27.

6 Conclusion

In this paper, we proposed the related-key boomerang attack to 174, 145 and 130 rounds of KATAN32/48/64, respectively, which dramatically improved the number of attacked rounds compared with the previous results. Examples of the right quartet on KATAN32 confirmed the feasibility of our attack. As far as we know, this is the best result achieved on the KATAN family.

References

1. De Cannière, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN — A Family of Small and Efficient Hardware-Oriented Block Ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009)
2. Bogdanov, A., Rechberger, C.: A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 229–240. Springer, Heidelberg (2011)
3. Wei, L., Rechberger, C., Guo, J., Wu, H., Wang, H., Ling, S.: Improved Meet-in-the-Middle Cryptanalysis of KTANTAN (Poster). In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 433–438. Springer, Heidelberg (2011)

4. Ågren, M.: Some Instant- and Practical-Time Related-Key Attacks on KTAN-TAN32/48/64. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 213–229. Springer, Heidelberg (2012)
5. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 130–145. Springer, Heidelberg (2010)
6. Isobe, T., Shibutani, K.: All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 202–221. Springer, Heidelberg (2013)
7. Albrecht, M.R., Leander, G.: An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 1–15. Springer, Heidelberg (2013)
8. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional Differential Cryptanalysis of Trivium and KATAN. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 200–212. Springer, Heidelberg (2012)
9. Knellwolf, S.: Accelerated Key Search for the KATAN Family of Block Ciphers. In: ECRYPT Workshop on Lightweight Cryptography (2011)
10. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 507–525. Springer, Heidelberg (2005)
11. Hong, S., Kim, J., Lee, S., Preneel, B.: Related-Key Rectangle Attacks on Reduced Versions of SHACAL-1 and AES-192. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 368–383. Springer, Heidelberg (2005)
12. Kim, J., Kim, G., Hong, S., Lee, S., Hong, D.: The Related-Key Rectangle Attack – Application to SHACAL-1. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 123–136. Springer, Heidelberg (2004)
13. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999)
14. Biham, E.: New Types of Cryptanalytic Attacks Using Related Keys. J. Cryptology 7(4), 229–246 (1994)
15. Kelsey, J., Schneier, B., Wagner, D.: Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 237–251. Springer, Heidelberg (1996)
16. Biham, E., Dunkelman, O., Keller, N.: A Unified Approach to Related-Key Attacks. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 73–96. Springer, Heidelberg (2008)
17. Kelsey, J., Kohno, T., Schneier, B.: Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 75–93. Springer, Heidelberg (2001)
18. Biham, E., Dunkelman, O., Keller, N.: The Rectangle Attack - Rectangling the Serpent. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 340–357. Springer, Heidelberg (2001)
19. Aoki, K., Sasaki, Y.: Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 70–89. Springer, Heidelberg (2009)
20. Lu, J., Kim, J., Keller, N., Dunkelman, O.: Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 370–386. Springer, Heidelberg (2008)

Appendix

Table 10. Example of confirmed boomerang quartets for KATAN32

P_1	0x46ec3236	C_1	0xee39e8a1	K_1	0x22fe640869975423bce9
P_2	0x4eed3216	C_2	0xf19133e1	K_2	0x22fe640869975c23bde9
P_3	0xd2379460	C_3	0xee11e925	K_3	0xa6ffe4826d8d3228d6c1
P_4	0xda369440	C_4	0xf1b93265	K_4	0xa6ffe4826d8d3a28d7c1
$P_1 \oplus P_2$	0x08010020	$C_1 \oplus C_3$	0x00280184	$K_1 \oplus K_2 = K_3 \oplus K_4$	0x0000000000008000100
$P_3 \oplus P_4$	0x08010020	$C_2 \oplus C_4$	0x00280184	$K_1 \oplus K_3 = K_2 \oplus K_4$	0x8401808a041a660b6a28

Table 11. Differential characteristic of KATAN32 E_0 (1 - 70)

Round	$L_2[0] \dots L_2[18]$	$L_1[0] \dots L_1[13]$	K_a	K_b	Pr.
0	000001000000000100	0000000010000	0	0	1
1	000000100000000010	0000000001000	0	0	2^{-1}
2	000000010000000001	0000000000100	0	0	2^{-1}
3	000000001000000000	0000000000010	0	0	2^{-1}
4	000000000100000000	0000000000001	1	0	2^{-2}
5	000000000010000000	0000000000000	0	0	2^{-2}
6	000000000001000000	0000000000000	0	0	2^{-3}
7	000000000000100000	0000000000000	0	0	2^{-3}
8	000000000000010000	0000000000000	0	0	2^{-4}
9	000000000000001000	0000000000000	0	0	2^{-4}
10	000000000000000100	0000000000000	0	0	2^{-4}
11	000000000000000010	0000000000000	0	0	2^{-4}
12	000000000000000001	0000000000000	0	0	2^{-4}
13	000000000000000000	0000000000000	0	1	2^{-4}
14	000000000000000000	0000000000000	0	0	2^{-4}
53	000000000000000000	0000000000000	0	1	2^{-4}
54	000000000000000000	1000000000000	0	0	2^{-4}
55	000000000000000000	0100000000000	0	0	2^{-4}
56	000000000000000000	0010000000000	0	0	2^{-4}
57	000000000000000000	0001000000000	0	0	2^{-4}
58	000000000000000000	0000100000000	0	0	2^{-4}
59	000000000000000000	0000010000000	0	0	2^{-4}
60	000000000000000000	0000001000000	1	0	2^{-5}
61	100000000000000000	0000000100000	0	0	2^{-5}
62	110000000000000000	0000000010000	0	0	2^{-5}
63	011000000000000000	0000000001000	0	0	2^{-6}
64	001100000000000000	0000000000100	0	0	2^{-6}
65	000110000000000000	0000000000010	0	0	2^{-7}
66	000011000000000000	0000000000001	0	1	2^{-8}
67	100001100000000000	1000000000000	0	0	2^{-8}
68	010000110000000000	0100000000000	0	0	2^{-8}
69	001000011000000000	1010000000000	0	0	2^{-8}
70	000100001100000000	1101000000000	0	0	2^{-9}

Table 12. Differential characteristic of KATAN32 E_1 (71 - 140)

Round	$L_2[0] \dots L_2[18]$	$L_1[0] \dots L_1[13]$	K_a	K_b	Pr.
70	000010000000000001000	0000000100001	0	0	1
71	000001000000000000100	0000000010000	0	0	1
72	000000100000000000010	0000000001000	0	0	2^{-1}
73	000000010000000000001	0000000000100	0	0	2^{-1}
74	000000001000000000000	0000000000010	0	0	2^{-1}
75	000000000100000000000	0000000000001	1	0	2^{-2}
76	000000000010000000000	0000000000000	0	0	2^{-2}
77	000000000001000000000	0000000000000	0	0	2^{-3}
78	000000000000100000000	0000000000000	0	0	2^{-3}
79	000000000000010000000	0000000000000	0	0	2^{-4}
80	000000000000001000000	0000000000000	0	0	2^{-4}
81	000000000000000100000	0000000000000	0	0	2^{-4}
82	000000000000000010000	0000000000000	0	0	2^{-4}
83	000000000000000001000	0000000000000	0	0	2^{-4}
84	000000000000000000100	0000000000000	0	1	2^{-4}
124	000000000000000000000	0000000000000	0	1	2^{-2}
125	000000000000000000000	1000000000000	0	0	2^{-2}
126	000000000000000000000	0100000000000	0	0	2^{-3}
127	000000000000000000000	0010000000000	0	0	2^{-3}
128	000000000000000000000	0001000000000	0	0	2^{-3}
129	000000000000000000000	0000100000000	0	0	2^{-4}
130	000000000000000000000	0000010000000	0	0	2^{-4}
131	000000000000000000000	0000001000000	1	0	2^{-5}
132	100000000000000000000	0000000100000	0	0	2^{-5}
133	110000000000000000000	0000000010000	0	0	2^{-5}
134	011000000000000000000	0000000001000	0	0	2^{-6}
135	001100000000000000000	0000000000100	0	0	2^{-6}
136	000110000000000000000	0000000000010	0	0	2^{-7}
137	000011000000000000000	0000000000001	0	1	2^{-8}
138	100001100000000000000	1000000000000	0	0	2^{-8}
139	010000110000000000000	0100000000000	0	0	2^{-8}
140	001000011000000000000	1010000000000	0	0	2^{-8}

Table 13. Differential characteristic of KATAN48 E_0 (1 - 60)

Round	$L_2(L_2[0]\ldots L_2[28])$	$L_1(L_1[0]\ldots L_1[18])$	K_a	K_b	Pr.
0	000000000110000000110000000011	000000000000000000011	1	0	1
1	000000000001100000001100000000	000000000000000000000	0	0	1
2	000000000000011000000011000000	000000000000000000000	0	0	2^{-2}
3	000000000000000110000000110000	000000000000000000000	0	0	2^{-4}
4	000000000000000001100000000110	000000000000000000000	0	0	2^{-5}
5	000000000000000000011000000001	000000000000000000000	0	0	2^{-5}
6	000000000000000000001100000000	000000000000000000000	0	0	2^{-6}
7	000000000000000000000000110000	000000000000000000000	0	0	2^{-7}
8	000000000000000000000000001100	000000000000000000000	0	0	2^{-7}
9	000000000000000000000000000011	000000000000000000000	0	1	2^{-7}
10	000000000000000000000000000000	000000000000000000000	0	0	2^{-7}
49	000000000000000000000000000000	000000000000000000000	0	1	2^{-7}
50	000000000000000000000000000000	110000000000000000000	0	0	2^{-7}
51	000000000000000000000000000000	001100000000000000000	0	0	2^{-7}
52	000000000000000000000000000000	000011000000000000000	0	0	2^{-7}
53	100000000000000000000000000000	000000110000000000000	0	0	2^{-7}
54	001000000000000000000000000000	000000001100000000000	0	0	2^{-9}
55	000010000000000000000000000000	000000000011000000000	0	0	2^{-9}
56	100000100000000000000000000000	000000000001100000000	1	0	2^{-9}
57	101000001000000000000000000000	000000000000011000000	0	0	2^{-10}
58	001010000010000000000000000000	000000000000000110000	0	0	2^{-12}
59	100010100000100000000000000000	000000000000000001000	0	0	2^{-12}
60	011000101000001000000000000000	000000000000000000000	0	0	2^{-14}

Table 14. Differential characteristic of KATAN48 E_1 (61 - 119)

Round	$L_2(L_2[0]\ldots L_2[28])$	$L_1(L_1[0]\ldots L_1[18])$	K_a	K_b	Pr.
60	000000000110000000110000000011	000000000000000000011	1	0	1
61	000000000001100000001100000000	000000000000000000000	0	0	1
62	000000000000011000000011000000	000000000000000000000	0	0	2^{-2}
63	000000000000000110000000110000	000000000000000000000	0	0	2^{-4}
64	000000000000000001100000000110	000000000000000000000	0	0	2^{-5}
65	000000000000000000011000000001	000000000000000000000	0	0	2^{-5}
66	000000000000000000001100000000	000000000000000000000	0	0	2^{-6}
67	000000000000000000000000110000	000000000000000000000	0	0	2^{-7}
68	000000000000000000000000001100	000000000000000000000	0	0	2^{-7}
69	000000000000000000000000000011	000000000000000000000	0	1	2^{-7}
70	000000000000000000000000000000	000000000000000000000	0	0	2^{-7}
109	000000000000000000000000000000	000000000000000000000	0	1	2^{-7}
110	000000000000000000000000000000	110000000000000000000	0	0	2^{-7}
111	000000000000000000000000000000	001100000000000000000	0	0	2^{-7}
112	000000000000000000000000000000	000011000000000000000	0	0	2^{-7}
113	100000000000000000000000000000	000000110000000000000	0	0	2^{-7}
114	001000000000000000000000000000	000000001100000000000	0	0	2^{-9}
115	000010000000000000000000000000	000000000011000000000	0	0	2^{-9}
116	100000100000000000000000000000	000000000001100000000	1	0	2^{9}
117	101000001000000000000000000000	000000000000011000000	0	0	2^{-10}
118	001010000010000000000000000000	000000000000000110000	0	0	2^{-12}
119	100010100000100000000000000000	000000000000000001000	0	0	2^{-12}

Table 15. Differential characteristic of KATAN64 E_0 (1 - 56)

Round	$L_2(L_2[0]\dots L_2[38])$	$L_1(L_1[0]\dots L_1[24])$	K_a	K_b	Pr.
0	000000000000000000011100000000011100000	0000000000000000000000000	0	0	1
1	000000000000000000000111000000000011100	0000000000000000000000000	0	0	2^{-3}
2	000000000000000000000000111000000000011	0000000000000000000000000	0	0	2^{-4}
3	000000000000000000000000001110000000000	0000000000000000000000000	0	0	2^{-4}
4	000000000000000000000000000000111000000	0000000000000000000000000	0	0	2^{-4}
5	000000000000000000000000000000000111000	0000000000000000000000000	0	0	2^{-6}
6	000000000000000000000000000000000000111	0000000000000000000000000	0	1	2^{-7}
7	000000000000000000000000000000000000000	0000000000000000000000000	0	0	2^{-7}
46	000000000000000000000000000000000000000	0000000000000000000000000	0	1	2^{-7}
47	000000000000000000000000000000000000000	1110000000000000000000000	0	0	2^{-7}
48	000000000000000000000000000000000000000	0001110000000000000000000	0	0	2^{-7}
49	000000000000000000000000000000000000000	0000001110000000000000000	0	0	2^{-7}
50	000000000000000000000000000000000000000	0000000001110000000000000	0	0	2^{-7}
51	000000000000000000000000000000000000000	0000000000001110000000000	0	0	2^{-10}
52	110000000000000000000000000000000000000	0000000000000001110000000	0	0	2^{-10}
53	001110000000000000000000000000000000000	0000000000000000001110000	1	0	2^{-10}
54	111001110000000000000000000000000000000	0000000000000000000001110	0	0	2^{-13}
55	110111001110000000000000000000000000000	0000000000000000000000001	0	0	2^{-14}
56	001110111001110000000000000000000000000	0000000000000000000000000	0	0	2^{-16}

Table 16. Differential characteristic of KATAN64 E_1 (57 - 113)

Round	$L_2(L_2[0]\dots L_2[38])$	$L_1(L_1[0]\dots L_1[24])$	K_a	K_b	Pr.
56	000000000000001110000000001110000000	0000000000000000000000000	0	0	1
57	000000000000000001110000000011100000	0000000000000000000000000	0	0	1
58	000000000000000000001110000000011100	0000000000000000000000000	0	0	2^{-3}
59	000000000000000000000011100000000011	0000000000000000000000000	0	0	2^{-4}
60	000000000000000000000000111000000000	0000000000000000000000000	0	0	2^{-4}
61	000000000000000000000000000111000000	0000000000000000000000000	0	0	2^{-4}
62	000000000000000000000000000000111000	0000000000000000000000000	0	0	2^{-6}
63	000000000000000000000000000000000111	0000000000000000000000000	0	1	2^{-7}
64	000000000000000000000000000000000000	0000000000000000000000000	0	0	2^{-7}
103	000000000000000000000000000000000000	0000000000000000000000000	0	1	2^{-7}
104	000000000000000000000000000000000000	1110000000000000000000000	0	0	2^{-7}
105	000000000000000000000000000000000000	0001110000000000000000000	0	0	2^{-7}
106	000000000000000000000000000000000000	0000001110000000000000000	0	0	2^{-7}
107	000000000000000000000000000000000000	0000000001110000000000000	0	0	2^{-7}
108	000000000000000000000000000000000000	0000000000001110000000000	0	0	2^{-10}
109	110000000000000000000000000000000000	0000000000000001110000000	0	0	2^{-10}
110	001110000000000000000000000000000000	0000000000000000001110000	1	0	2^{-10}
111	111001110000000000000000000000000000	0000000000000000000001110	0	0	2^{-13}
112	110111001110000000000000000000000000	0000000000000000000000001	0	0	2^{-14}
113	001110111001110000000000000000000000	0000000000000000000000000	0	0	2^{-16}