Improved Conditional Differential Analysis on NLFSR Based Block Cipher KATAN32 with MILP

Zhaohui Xing[1,2], Wenying Zhang[1(✉)], and Guoyong Han[3]

[1] School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China
3613452@qq.com, zhangwenying@sdnu.edu.cn
[2] School of Science, Shandong Jiaotong University, Jinan 250357, China
[3] School of Management Engineering, Shandong Jianzhu University, Jinan 250101, China
hgy_126@126.com

Abstract. This paper describes constructing a Mixed Integer Linear Programming (MILP) model for conditional differential cryptanalysis on nonlinear feedback shift register (NLFSR)-based block ciphers, and proposes an approach for detecting the bit with a strongly-biased difference. The model is successfully applied to the block cipher KATAN32 in the single-key scenario, resulting in practical key-recovery attacks covering more rounds than the previous. In particular, we present two distinguishers for 79 and 81 out of 254 rounds of KATAN32. Based on the 81-round distinguisher we recover 11 equivalent key bits of 98-round KATAN32 with the time complexity being less than 2^{31} encryptions of 98-round KATAN32 and recover 13 equivalent key bits of 99-round KATAN32 with the time complexity being less than 2^{33} encryptions of 99-round KATAN32. Thus far, our results are the best known practical key-recovery attacks for the round-reduced variants of KATAN32 as far as the number of rounds and the time complexity. All the results are verified experimentally.

Keywords: KATAN block cipher · Conditional differential cryptanalysis · Mixed Integer Linear Programming (MILP)

1 Introduction

Radio frequency identification (RFID) technology has been used in many aspects of life such as access control, parking management, identification, goods tracking, and wireless sensor networks (WSNs) have been used for various critical industrial applications, such as heart beat monitoring, temperature monitoring for precision agriculture, and power usage monitoring for smart grid [22]. In this new cryptography environment, applications of RFID technology and sensor networks have similar features such as weak computation ability, small

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2021
Published by Springer Nature Switzerland AG 2021. All Rights Reserved
D. Wang et al. (Eds.): SPNCE 2020, LNICST 344, pp. 370–393, 2021.
https://doi.org/10.1007/978-3-030-66922-5_26

storage space, and strict power constraints, and such very constrained environments require new cryptographic primitives such as tiny and efficient ciphers. This tends to make traditional block ciphers such as AES not suitable for these constrained environments. To meet these needs, a number of lightweight ciphers, including PRESENT [5], KATAN and KTANTAN family [6], have been proposed.

The KATAN and KTANTAN block ciphers were proposed by Christophe DeCannière, Orr Dunkelman and Miroslav Knezevic at CHES 2009 [6]. To increase its speed, KATAN uses nonlinear feedback shift registers (NLFSRs) as well as linear key schedule. Both KATAN and KTANTAN have three variants with 32-bit, 48-bit and 64-bit block sizes, each requiring an 80-bit user key. In addition, KATAN and KTANTAN share the same data path specification, including round transformation and round constants. The only difference between KATAN and KTANTAN is the generation of subkeys. For KTANTAN, two bits of the 80-bit $K = k_{79}k_{78}...k_1k_0$ are selected each round. But the key schedule of the KATAN32 cipher (and the other two variants KATAN48 and KATAN64) loads the 80-bit key into an LFSR (the least significant bit of the key is loaded to position 0 of the LFSR), and for each round positions 0 and 1 of the LFSR are generated as the round subkey k_{2i} and k_{2i+1}, and the LFSR is clocked twice. Because of the simple key schedule, KTANTAN was broken by Wei et al. [23], and while a more complex key schedule makes KATAN secure and stronger, the key schedule is also linear.

1.1 Related Work

KATAN family ciphers have been analyzed by extensive cryptanalysis. At ASIACRYPT 2010, Knellwolf et al. analyzed KATAN&KTANTAN [14] using conditional differential cryptanalysis [3] and recovered four equivalent key bits for 78 of 254 rounds of KATAN32 in the single-key scenario. They subsequently analyzed KATAN32 in the related-key scenario with an improved technique using automatic tools, then obtained key-recovery attacks for 120 of 254 rounds of KATAN32 [15]. Finding the non-uniformity of the difference distribution after 91 rounds, Martin R. Albrecht and Gregor Leander proposed an 91-round distinguisher with the time complexity being 2^{32} encryptions [2]. These results on KATAN32 are listed in Table 1.

Other types of attacks formally published on this cipher, such as all subkeys recovery (ASR) which is a variant of the meet-in-the-middle (MITM) attack [12], Match Box MITM attack [9], Dynamic Cube attack [1], Multidimensional MITM attack [18,28], are also listed in Table 1. As can be seen from the details in Table 1, each time complexity is too high to present a practical attack.

As stated in [11], related-key attacks are arguable in a practical sense, because a related-key attack is under the assumption that the attacker had known and even controlled the relation between multiple unknown keys. Because of this assumption, the related-key attack is arguable from the aspect of practical security, though it is meaningful during the design and certification of a cipher. In particular, the key of an ultra lightweight block cipher in low-end devices such

Table 1. Cryptanalytic results on KATAN32

Type	Scenario	Rounds	Time complexity	Reference
Conditional differential	Single-key	78	2^{22}	[14]
	Related-key	120	2^{31}	[15]
Improved conditional differential	Single-key	97	2^{30}	This paper
	Single-key	98	2^{31}	This paper
	Single-key	99	2^{33}	This paper
Differential	Single-key	91	2^{32}*	[2]
	Single-key	115	2^{78}	
MIMT ASR	Single-key	119	$2^{79.1}$	[12]
Match box MITM	Single-key	153	$2^{78.5}$	[9]
Dynamic cube	Single-key	155	$2^{78.3}$	[1]
Multidimensional MITM	Single-key	175	$2^{79.3}$	[28]
	Single-key	206	2^{79}	[18]

*A 91-round differential distinguisher

as a passive RFID tag may not be changed during its life cycle so, in a practical sense, the security of a lightweight cipher under the single-key scenario is the most important. As shown in [13], even though the result of an attack in the related-key scenario is better, it is still meaningful to explore an attack in the single-key scenario.

Conditional differential cryptanalysis was first introduced by Biham and Ben-Aroya at Crypto 1993 [3], since this technique, which is a very popular technique in hash functions cryptanalysis, allows increasing the probability of a differential characteristic being satisfied under some conditions, it also can be useful for block ciphers. Mixed integer linear programming (MILP) is a general mathematical tool for optimization that takes as inputs a linear objective function and a system of linear inequalities and finds solutions that optimize the objective function under the constraints of all inequalities. It was first applied by Mouha et al. in [17] and Wu in et al. [25] to count the active Sboxes of word-based block ciphers, and Sun et al. in [21] used it to search for differential characteristics and linear approximations. It has been also applied to search for integral distinguishers and division trails [26] and impossible differentials [19]. In particular, it has been applied to key-recovery attacks of keyed Keccak MAC, where attackers implemented conditional cube attacks on Keccak with the propagation of cube variables controlled under conditions in the first several rounds and attacked keyed Keccak [10,16,20].

1.2 Our Contributions

In this paper, we improved conditional differential attacks from two aspects. On the one hand, we propose a method of automatic conditional differential cryptanalysis using MILP. This method helps us minimize the number of conditions

under which the differential characteristic can hold, because of the fewer the conditions, the higher the probability of the differential path. On the other hand, we propose a method to calculate the bias of every bit quickly and detect the bit which has a strongly biased difference. Finally, using the standard differential attack, we extended the conditional differential attack to more rounds. The details are described in the following paragraphs.

We first propose a novel method using MILP to automatically search an initial difference and conditions for conditional differential cryptanalysis. In [14], Knellwolf chose initial differences manually and it is difficult to find the optimal choice, a crucial element in this attack. In this paper, we solve this problem using MILP. We analyze how to identify conditions on internal state variables, and then, by modeling relations between differences in state bits and conditions, we construct a linear inequality system. The object function of this MILP problem is the minimum number of conditions in a certain number rounds. Based on the method using MILP, we automatically obtain the initial difference and conditions.

Second, we present an approach to detect the bias in the difference of the update bit. In [14], Knellwolf detected the bias experimentally by observing certain non-randomness of a difference of the update bit. We find that the probability of a difference in the update bit is determined by the probabilities of differences in bits that generate the update bit. After the analysis we present a formula for evaluating the probability of the difference in the update bit, helping us detect which bit has a strongly biased difference.

Given the initial difference, the conditions, and the position of the bit with a bias, we are able to mount a key-recovery attack.

We apply conditional differential cryptanalysis with these two improvements to analyze the security of KATAN32. It is shown that we can retrieve ten equivalent key bits for the variant of KATAN32 with 79 initialization rounds and four equivalent key bits with 81 initialization rounds.

Using standard differential attacks, we extend the 81-round conditional differential key-recovery attacks to 97-round, 98-round and 99-round with time complexity being 2^{30}, 2^{31}, 2^{33} encryptions respectively. Extended key-recovery attacks can recover 10,11 and 13 equivalent key bits respectively. This is the best known practical cryptanalytic result on KATAN32 so far.

All of our attacks succeed experimentally. All of our source codes and experiments results are available at https://www.dropbox.com/sh/028s4f06f363b2h/AADItFkz-N1KaAMZR7nIPTawa?dl=0.

1.3 Organization

The paper is organized as follows. In Sect. 2, some preliminaries are introduced. Section 3 describes the two improvements in conditional differential attacks. In Sect. 4, with these improvements, the attacks mounted on 79 and 81 of 254 rounds of KATAN32 are presented in detail. In Sect. 5 we extend the attacks to 97, 98 and 99 of 254 rounds of KATAN32 combined with standard differential attacks. Finally, we conclude the paper in Sect. 6.

2 Preliminaries

We present our notations in Table 2.

Table 2. The notations used throughout the paper

Symbol	Definition
F_2	The finite field of two elements
F_2^n	The n-dimensional vector space over F_2
S_t	The state of the 13-bit NLFSR at round t
L_t	The state of the 19-bit NLFSR at round t
s_{t+i}	The i-th bit of the 13-bit NLFSR at round t
l_{t+i}	The i-th bit of the 19-bit NLFSR at round t
Δs_{t+i}	The difference in s_{t+i}
Δl_{t+i}	The difference in l_{t+i}
X	A 32-bit plaintext block
x_i	The i-th bit of the plaintext
K	An 80-bit key
k_i	The i-th bit of the key

2.1 Description of KATAN

KATAN and KTANTAN are composed of three block ciphers with 32, 48, 64-bit block sizes, respectively, denoted by KATANn and KTANTANn for $n = 32, 48, 64$. These 6 ciphers all have 80-bit keys and the only difference between KATAN and KTANTAN is the key schedule.

Here we will briefly describe KATAN32, which is analyzed in this paper.

Key Schedule. The master key $K = (k_0, \cdots, k_{79})$ is loaded into an 80-bit linear feedback register and new round keys are generated by the linear feedback relation:

$$k_{i+80} = k_i \oplus k_{i+19} \oplus k_{i+30} \oplus k_{i+67}, 0 \leq i \leq 427.$$

In the reminder of this paper, for any $i \geq 80$, we call k_i one equivalent key bit.

Round Function. In initialization, a 32-bit plaintext block $X=(x_{31}, \ldots, x_0)$ is loaded into two NLFSRs with lengths 13 and 19 bits, respectively. Denote states of the 13-bit NLFSR and the 19-bit NLFSR at round t as $S_t = (s_t, s_{t+1}, \ldots, s_{t+12})$ and $L_t = (l_t, l_{t+1}, \ldots, l_{t+18})$.

When $t = 0$, the plaintext is loaded as $l_{t+i} = x_{18-i}$ for $0 \leq i \leq 18$ and $s_{t+i} = x_{31-i}$ for $0 \leq i \leq 12$.

At round t, for $0 \leq t \leq 253$, two new bits s_{t+13} and l_{t+19} are produced according to following equations:

$$s_{t+13} = l_t \oplus l_{t+11} \oplus l_{t+6}l_{t+8} \oplus l_{t+10}l_{t+15} \oplus k_{2t+1}, \tag{1}$$

$$l_{t+19} = s_t \oplus s_{t+5} \oplus s_{t+4}s_{t+7} \oplus s_{t+9}a_t \oplus k_{2t}, \tag{2}$$

where a_t is a round constant generated by the 8-bit LFSR using the recursive relation $a_t = a_{t-3} \oplus a_{t-5} \oplus a_{t-7} \oplus a_{t-8}(t \geq 8)$ with the seed value $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7) = (1, 1, 1, 1, 1, 1, 1, 0)$. After 254 rounds, the state is outputted as the ciphertext. The round function is depicted in Fig. 1.

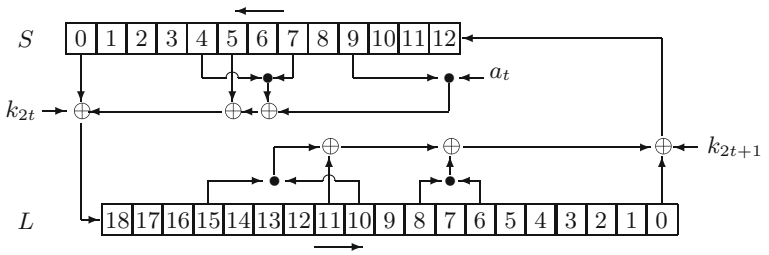

Fig. 1. The round function of KATAN32 cipher

2.2 Conditional Differential Analysis

Knellwolf et al. applied conditional differential cryptanalysis to NLFSR-based cryptosystems at ASIACRYPT 2010 [14]. This technique is based on differential cryptanalysis used to analyze initialization mechanisms of stream ciphers in [4,7,24]. After choosing an initial difference, it studies the propagation of an input difference through NLFSR-based cryptosystems and identifies conditions on internal state bits to prevent propagation whenever possible. By taking the plaintext pairs conforming with these conditions as input, biases can be detected in differences of update bits at some rounds. Once a bias is detected, the key is considered to obey the expected conditions and we obtain information for secret key bits. In some cases where there are single key bits and relations of key bits in the conditions, we call each of them one equivalent key bit, leading to a key-recovery attack.

3 Improved Conditional Differential Cryptanalysis

In [14], authors traced differences through cryptosystems and prevented their propagation whenever possible by identifying conditions on internal state variables. They gave suggestions on how to manually choose an initial difference rather than a specific method for acquiring it. These suggestions were that the

difference propagation should be controllable for as many rounds as possible with a small number of conditions and that there should not be too many conditions involving bits of K during initial rounds. While the initial difference is of crucial importance with respect to the number of rounds attacked, it is not easy to manually choose a suitable initial difference. In this paper, we propose a novel method using MILP to search for an initial difference, deriving as few conditions as possible and the differential characteristic that covers as many rounds as possible. We also present a method for evaluating the probability of the difference in the update bit, by which we can detect the bit with an obvious bias.

Using these two improvements, we apply the improved conditional differential cryptanalysis to NLFSR based block cipher KATAN32. The framework of the analysis is divided into the following four steps.

Search for an Initial Difference with MILP. With the method described in Subsect. 3.1, one can formulate an MILP model of difference propagation during each round, search for a differential characteristic with minimum conditions, and obtain the initial difference of the characteristic and conditions.

Choose Conditions. We trace the propagation of the initial difference and identify conditions that prevent propagation of differences until the number of key bits and plaintext bits involved in conditions becomes too great(exceed the enumeration capability) to mount an attack.

Calculate the Bias. Given the initial difference and conditions chosen in the previous steps, the probability of the difference in each bit of the two NLFSR at the round when the conditions cease being applied can be easily derived. Taking this probability as input of the method described in Subsect. 3.2, we can calculate the probability of the difference in update bit at each subsequent round. According to these probabilities, we can locate the bit whose difference has an obvious bias and number of rounds is largest.

Mount the Key-Recovery Attack. Since the conditions include some equivalent key bits, if plaintexts are selected with the conditions consisting of correct equivalent key bits, the difference in the update bit located during the previous step would show the bias. Utilizing conditions and the bit with the bias, equivalent key bits involved in the conditions can be recovered. The attack is involved in Algorithm 1.

3.1 Modeling the Difference Propagation of the Round Function

By modeling the propagation of differences under the control of conditions that can prevent the generation of a difference in the update bit, we obtain a conditional differential characteristic with the fewest conditions, and we can also obtain the initial difference and conditions applied from this conditional differential characteristic. The steps are as follows:

Algorithm 1: The framework of the conditional differential attack

Input: Equation (1) and Equation (2) ;
Output: g: correct equivalent key bits;
Obtain an initial difference ΔX and a conditions set κ by MILP technique;
$\kappa \leftarrow$ {conditions chosen from κ in the previous rounds to make sure that the number of key bits and plaintext bits involved in conditions should not exceed the enumeration capability} ;
$\lambda \leftarrow$ {the probability of the difference of each bit at round r from which conditions just cease being applied. It is derived from ΔX and κ};
$P \leftarrow$ {the probabilities of the differences of each subsequent update bits after round r calculated by λ using the method described in subsection 3.2 };
$t \leftarrow$ the bit derived from P having the non-zero bias and at the highest possible number of rounds;
for $g \in$ {*enumerate equivalent key bits involved in* κ} **do**
  $count1 = 0$;
  $count0 = 0$;
  for $x \in$ {*enumerate plaintext bits involved in* κ} **do**
    if x, g *satisfy* κ **then**
      calculate Δt from x and ΔX;
      if $\Delta t = 1$ **then**
        $count1++$;
      else
        $count0++$;
      end
    end
  end
  $P\{\Delta t = 1\} = count1/(count1 + count0)$;
  $A \leftarrow A \cup \{(g, |P\{\Delta t = 1\} - \frac{1}{2}|)\}$;
end
searching in A for the max $|P\{\Delta t = 1\} - \frac{1}{2}|$;
return g in accordance with the max $|P\{\Delta t = 1\} - \frac{1}{2}|$;

1. Finding all Modes of Difference Propagation Under the Control of Conditions. For KATAN32 based on NLFSR, at each round only two bits are generated by some bits from the previous round, so the differences in these two update bits are caused only by the bits involved in Eq. (1) and Eq. (2).

There are linear and non-linear terms in Eq. (1) and Eq. (2). If there are differences in non-linear terms, the difference in the update bit can be canceled by imposing conditions even if there are at the same time differences in linear terms. If differences appear only in linear terms, there are no possible conditions that could be applied to cancel the differences; they only can be canceled by one another or they propagate to the next round.

For example, for Eq. (1): $s_{t+13} = l_t \oplus l_{t+11} \oplus l_{t+6}l_{t+8} \oplus l_{t+10}l_{t+15} \oplus k_{2t+1}$, if $\Delta l_{t+6} = 1$, with the other bits having no differences, we add the condition $l_{t+8} = 0$ to ensure that $\Delta s_{t+13} = 0$. The number of conditions is 1 and the difference of the update bit is 0. If $\Delta l_{t+11} = 1$, with the other bits have no

differences, no conditions could cancel the difference which appears in the update bit and propagates to the next round. In this case, the number of conditions is 0 and the difference of the update bit is 1.

This shows that we can apply conditions to prevent the propagation of differences when the difference state (we call the difference of the internal state the difference state) is at some particular value, while for some other values there are no conditions that could prevent the propagation of the differences.

For each exact difference state, it can be confirmed whether conditions could be applied and whether there would be a difference in the update bit according to the previous strategy that aimed at preventing the propagation of differences.

With respect to Eq. (1), s_{t+13} is generated by 6 bits in the 19-bit NLFSR of round t, so that the difference of s_{t+13} depends on the values and the differences of these 6 bits. Let c (the flag of adding a condition) denote whether a condition is applied to cancel the difference of the update bit, and let us search all values of the vector (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}, Δs_{t+13},c) following the following strategies.

If $\Delta s_{t+13} = 0$ according to Eq. (1), c takes value 0.

Example 1. If (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}) = (1, 1, 0, 0, 0, 0), according to Eq. (1), $\Delta s_{t+13} = 0$. Since no conditions need to be added, (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}, Δs_{t+13}, c) = (1, 1, 0, 0, 0, 0, 0, 0) is the vector we hunt.

Assuming that Δs_{t+13} may be 1 or 0 according to Eq. (1). If a condition could be applied to ensure that $\Delta s_{t+13} = 0$, Δs_{t+13} takes value 0 and c takes value 1.

Example 2. Suppose that (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}) = (0, 0, 0, 0, 0, 1), according to Eq. (1), Δs_{t+13} could be either 1 or 0. But if we impose the condition $l_{t+10} = 0$, Δs_{t+13} must be 0, and c takes value 1, so (0,0,0,0,0,1,0,1) is the vector we hunt.

If there must be a difference in s_{t+13} and no conditions can cancel it, Δs_{t+13} takes value 1 and c takes value 0.

Example 3. Suppose that (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}) = (1, 0, 0, 0, 0, 0), according to Eq. (1) $\Delta s_{t+13} = 1$, and it cannot be canceled by any conditions. Then Δs_{t+13} takes value 1 and c takes value 0. So we obtain (1,0,0,0,0,0,1,0).

The difference state ($\Delta l_t, \Delta l_{t+11}$, $\Delta l_{t+6}, \Delta l_{t+8}$, Δl_{t+10}, Δl_{t+15}) can take on one of $2^6 = 64$ values. We derive the exact values of c and Δs_{t+13} from each value of the difference state in accordance with the above strategies. Then, with respect to Eq. (1), we get all 64 values of the 8-dimensional vector (Δl_t, Δl_{t+11}, Δl_{t+6}, Δl_{t+8}, Δl_{t+10}, Δl_{t+15}, Δs_{t+13}, c), presented in Table 3. Meanwhile, with respect to Eq. (2), we can also find all the difference state values (Δs_t, Δs_{t+5}, Δs_{t+4}, Δs_{t+7}, Δs_{t+9}, Δl_{t+19}, c). It should be noted that in Eq. (2) there is a constant a_t that achieves an exact value at each round. To simplify constraints of the MILP, we model two cases corresponding to the values of a_t.

Table 3. 64 vectors $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}, \Delta s_{t+13}, c)$

(0, 0, 0, 0, 0, 0, 0, 0)	(0, 1, 0, 1, 1, 0, 0, 1)	(1, 0, 1, 1, 0, 0, 0, 1)
(0, 0, 0, 0, 0, 1, 0, 1)	(0, 1, 0, 1, 1, 1, 0, 1)	(1, 0, 1, 1, 0, 1, 0, 1)
(0, 0, 0, 0, 1, 0, 0, 1)	(0, 1, 1, 0, 0, 0, 0, 1)	(1, 0, 1, 1, 1, 0, 0, 1)
(0, 0, 0, 0, 1, 1, 0, 1)	(0, 1, 1, 0, 0, 1, 0, 1)	(1, 0, 1, 1, 1, 1, 0, 1)
(0, 0, 0, 1, 0, 0, 0, 1)	(0, 1, 1, 0, 1, 0, 0, 1)	(1, 1, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 1, 0, 1, 0, 1)	(0, 1, 1, 0, 1, 1, 0, 1)	(1, 1, 0, 0, 0, 1, 0, 1)
(0, 0, 0, 1, 1, 0, 0, 1)	(0, 1, 1, 1, 0, 0, 0, 1)	(1, 1, 0, 0, 1, 0, 0, 1)
(0, 0, 0, 1, 1, 1, 0, 1)	(0, 1, 1, 1, 0, 1, 0, 1)	(1, 1, 0, 0, 1, 1, 0, 1)
(0, 0, 1, 0, 0, 0, 0, 1)	(0, 1, 1, 1, 1, 0, 0, 1)	(1, 1, 0, 1, 0, 0, 0, 1)
(0, 0, 1, 0, 0, 1, 0, 1)	(0, 1, 1, 1, 1, 1, 0, 1)	(1, 1, 0, 1, 0, 1, 0, 1)
(0, 0, 1, 0, 1, 0, 0, 1)	(1, 0, 0, 0, 0, 0, 1, 0)	(1, 1, 0, 1, 1, 0, 0, 1)
(0, 0, 1, 0, 1, 1, 0, 1)	(1, 0, 0, 0, 0, 1, 0, 1)	(1, 1, 0, 1, 1, 1, 0, 1)
(0, 0, 1, 1, 0, 0, 0, 1)	(1, 0, 0, 0, 1, 0, 0, 1)	(1, 1, 1, 0, 0, 0, 0, 1)
(0, 0, 1, 1, 0, 1, 0, 1)	(1, 0, 0, 0, 1, 1, 0, 1)	(1, 1, 1, 0, 0, 1, 0, 1)
(0, 0, 1, 1, 1, 0, 0, 1)	(1, 0, 0, 1, 0, 0, 0, 1)	(1, 1, 1, 0, 1, 0, 0, 1)
(0, 0, 1, 1, 1, 1, 0, 1)	(1, 0, 0, 1, 0, 1, 0, 1)	(1, 1, 1, 0, 1, 1, 0, 1)
(0, 1, 0, 0, 0, 0, 1, 0)	(1, 0, 0, 1, 1, 0, 0, 1)	(1, 1, 1, 1, 0, 0, 0, 1)
(0, 1, 0, 0, 0, 1, 0, 1)	(1, 0, 0, 1, 1, 1, 0, 1)	(1, 1, 1, 1, 0, 1, 0, 1)
(0, 1, 0, 0, 1, 0, 0, 1)	(1, 0, 1, 0, 0, 0, 0, 1)	(1, 1, 1, 1, 1, 0, 0, 1)
(0, 1, 0, 0, 1, 1, 0, 1)	(1, 0, 1, 0, 0, 1, 0, 1)	(1, 1, 1, 1, 1, 1, 0, 1)
(0, 1, 0, 1, 0, 0, 0, 1)	(1, 0, 1, 0, 1, 0, 0, 1)	
(0, 1, 0, 1, 0, 1, 0, 1)	(1, 0, 1, 0, 1, 1, 0, 1)	

When $a_t = 1$, Eq. (2) contains five boolean variables s_t, s_{t+5}, s_{t+4}, s_{t+7}, s_{t+9}, so the difference state (Δs_t, Δs_{t+5}, Δs_{t+4}, Δs_{t+7}, Δs_{t+9}) can take on one of $2^5 = 32$ different values deriving the 32 values of the 7-dimensional vector (Δs_t, Δs_{t+5}, Δs_{t+4}, Δs_{t+7}, Δs_{t+9}, Δl_{t+19}, c) shown in Table 4.

When $a_t = 0$, Eq. (2) contains four boolean variables s_t, s_{t+5}, s_{t+4}, s_{t+7}, so the difference state (Δs_t, Δs_{t+5}, Δs_{t+4}, Δs_{t+7}) can take on one of $2^4 = 16$ different values that lead to the 16 values of the 6-dimensional vector (Δs_t, Δs_{t+5}, Δs_{t+4}, Δs_{t+7}, Δl_{t+19}, c) shown in Table 5.

2. Modeling the Vector Sets Using Linear Inequalities. Via SageMath at http://www.sagemath.org, we obtain 19 linear inequalities that accurately describe the set of the 64 8-dimensional vectors in Table 3, that is, this set of linear inequalities characterize the difference propagation of Eq. (1) under the control of conditions. There are 10 inequalities remaining after a simple reduction. L_1 shows the 10 inequalities.

Table 4. 32 vectors $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta s_{t+9}, \Delta l_{t+19}, c)$

$(0,0,0,0,0,0,0)$	$(0,1,0,1,1,0,1)$	$(1,0,1,1,0,0,1)$
$(0,0,0,0,1,1,0)$	$(0,1,1,0,0,0,1)$	$(1,0,1,1,1,0,1)$
$(0,0,0,1,0,0,1)$	$(0,1,1,0,1,0,1)$	$(1,1,0,0,0,0,0)$
$(0,0,0,1,1,0,1)$	$(0,1,1,1,0,0,1)$	$(1,1,0,0,1,1,0)$
$(0,0,1,0,0,0,1)$	$(0,1,1,1,1,0,1)$	$(1,1,0,1,0,0,1)$
$(0,0,1,0,1,0,1)$	$(1,0,0,0,0,1,0)$	$(1,1,0,1,1,0,1)$
$(0,0,1,1,0,0,1)$	$(1,0,0,0,1,0,0)$	$(1,1,1,0,0,0,1)$
$(0,0,1,1,1,0,1)$	$(1,0,0,1,0,0,1)$	$(1,1,1,0,1,0,1)$
$(0,1,0,0,0,1,0)$	$(1,0,0,1,1,0,1)$	$(1,1,1,1,0,0,1)$
$(0,1,0,0,1,0,0)$	$(1,0,1,0,0,0,1)$	$(1,1,1,1,1,0,1)$
$(0,1,0,1,0,0,1)$	$(1,0,1,0,1,0,1)$	

Table 5. 16 vectors $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta l_{t+19}, c)$

$(0,0,0,0,0,0)$	$(0,1,1,0,0,1)$	$(1,1,0,0,0,0)$
$(0,0,0,1,0,1)$	$(0,1,1,1,0,1)$	$(1,1,0,1,0,1)$
$(0,0,1,0,0,1)$	$(1,0,0,0,1,0)$	$(1,1,1,0,0,1)$
$(0,0,1,1,0,1)$	$(1,0,0,1,0,1)$	$(1,1,1,1,0,1)$
$(0,1,0,0,1,0)$	$(1,0,1,0,0,1)$	
$(0,1,0,1,0,1)$	$(1,0,1,1,0,1)$	

$$
L_1 = \begin{cases}
-\Delta s_{t+13} - c + 1 \geq 0 \\
-\Delta l_{t+6} + c \geq 0 \\
-\Delta l_t - \Delta l_{t+11} - \Delta s_{t+13} + 2 \geq 0 \\
-\Delta l_{t+8} + c \geq 0 \\
-\Delta l_{t+10} + c \geq 0 \\
-\Delta l_{t+15} + c \geq 0 \\
\Delta l_t + \Delta l_{t+11} - \Delta s_{t+13} \geq 0 \\
-\Delta l_t + \Delta l_{t+11} + \Delta s_{t+13} + c \geq 0 \\
\Delta l_t - \Delta l_{t+11} + \Delta s_{t+13} + c \geq 0 \\
\Delta l_{t+6} + \Delta l_{t+8} + \Delta l_{t+10} + \Delta l_{t+15} - c \geq 0
\end{cases} \tag{3}
$$

Using the same method, we obtain two sets of linear inequalities L_2 and L_3 that accurately describe the 32 7-dimensional vectors given in Table 4 and the 16 6-dimensional vectors given in Table 5. The two sets are shown below:

$$L_2 = \begin{cases} -\Delta s_t - c + 1 \geq 0 \\ -\Delta s_{t+7} + c \geq 0 \\ -\Delta s_{t+4} + c \geq 0 \\ \Delta s_{t+4} + \Delta s_{t+7} - c \geq 0 \\ \Delta s_t - \Delta s_{t+5} - \Delta s_{t+9} - \Delta s_{t+2} \geq 0 \\ -\Delta s_t + \Delta s_{t+5} - \Delta s_{t+9} - \Delta s_{t+2} \geq 0 \\ -\Delta s_t - \Delta s_{t+5} + \Delta s_{t+9} - \Delta s_{t+2} \geq 0 \\ \Delta s_t + \Delta s_{t+5} + \Delta s_{t+9} - \Delta s_t \geq 0 \\ -\Delta s_t - \Delta s_{t+5} - \Delta s_{t+9} + \Delta s_t + c + 2 \geq 0 \\ \Delta s_t + \Delta s_{t+5} - \Delta s_{t+9} + \Delta s_t + c \geq 0 \\ \Delta s_t - \Delta s_{t+5} + \Delta s_{t+9} + \Delta s_t + c \geq 0 \\ -\Delta s_t + \Delta s_{t+5} + \Delta s_{t+9} + \Delta s_t + c \geq 0 \end{cases} \tag{4}$$

$$L_3 = \begin{cases} -\Delta s_t - c + 1 \geq 0 \\ -\Delta s_{t+4} + c \geq 0 \\ -\Delta s_{t+7} + c \geq 0 \\ \Delta s_{t+4} + \Delta s_{t+7} - c \geq 0 \\ \Delta s_t - \Delta s_{t+5} + \Delta s_t + c \geq 0 \\ -\Delta s_t + \Delta s_{t+5} + \Delta s_t + c \geq 0 \\ \Delta s_t + \Delta s_{t+5} - \Delta s_t \geq 0 \\ -\Delta s_t - \Delta s_{t+5} - \Delta s_{t+2} \geq 0 \end{cases} \tag{5}$$

3. Formulating the MILP Model to Determine an Initial Difference and Minimum Conditions. With these linear inequalities, we can obtain the relationships among the differences of bits that generate the update bit, the flag of adding a condition and the difference of the update bit in one round. We then expand the linear inequalities to n rounds, where n is a selected number, to obtain constraints of the MILP model. The objective function to be minimized is $\sum_{i=0}^{n} c_i$. The constraint of the initial difference is $\sum_{i=0}^{31} \Delta x_i \geq 1$. In our work, the MILP problem is solved by Cplex. With this solution, we can obtain both an initial difference and minimum conditions.

There are too many plaintext bits and key bits in the conditions applied in the later rounds, so we prefer applying the conditions in earlier rounds rather than all of them. No more conditions have been applied since a particular round, which leads to uncontrollable difference propagation in subsequent rounds. After several rounds, the probability of the difference in the update bit would always be 1/2. In Subsect. 3.2, we propose a method to evaluate the difference probability of the update bit, which helps us to find the bit whose difference probability deviates significantly from 1/2 and has the largest number of rounds, and utilize this bit to mount key-recovery attacks.

3.2 Detecting the Bias of the Difference

In [14,15], a bias was detected by experimentally observing certain non-randomness, and we now present a method for automatically detecting the bias by programming. The method produces a formula for calculating the probability of the update bit difference, enabling us to find the bit whose probability of the difference has a bias from 1/2. The greater the bias, the higher probability of a successful attack.

The properties below show that we can evaluate the probability of difference in the update bit, given all the probabilities of difference in the bits that generate the update bit. Thus, using the probability of difference in each bit of the two NLFSR at the round where the conditions cease being applied, we can calculate probabilities of update bits in each and every subsequent round.

Property 1. Let a, b be two independent random boolean variables, then the probability $P\{\Delta(a \oplus b) = 1\} = P\{\Delta a = 1\} + P\{\Delta b = 1\} - P\{\Delta a = 1\}P\{\Delta b = 1\}$.

With Property 1, if the probabilities of the differences in a and b were known, we could evaluate the probability of the difference in $a \oplus b$. It can be extended to the sum of four boolean variables:

Property 2. Let x, y, z, w be independent random boolean variables, then the probability

$$\begin{aligned} & P\{\Delta(x \oplus y \oplus z \oplus w) = 1\} \\ = & \frac{1}{2} - \frac{1}{2}(1 - 2P\{\Delta x = 1\})(1 - 2P\{\Delta y = 1\}) \\ & (1 - 2P\{\Delta z = 1\})(1 - 2P\{\Delta w = 1\}). \end{aligned}$$

In the following, we consider the case when the boolean variable where the difference is in is the product of the other two boolean variables. Property 3 shows me how to evaluate the probability.

Property 3. Let a, b be the same as defined in Property 1, then the probability $P\{\Delta(a \cdot b) = 1\} = (P\{\Delta a = 1\} + P\{\Delta b = 1\} - P\{\Delta a = 1\}P\{\Delta b = 1\}) \cdot \frac{1}{2}$.

In Eq. (1) and Eq. (2), since there is no difference in the key and const, k_{2t+1} , k_{2t} and a_t do not influence the probability of the difference. Accordingly, we can derive the results as follows:

From Eq. (1), we can obtain the formula to calculate the probability of $\Delta s_{t+13} = 1$:

$$\begin{aligned} & P\{\Delta s_{t+13} = 1\} \\ = & P\{\Delta(l_t \oplus l_{t+11} \oplus l_{t+6}l_{t+8} \oplus l_{t+10}l_{t+15}) = 1\} \\ = & \frac{1}{2} - \frac{1}{2} \cdot (1 - 2P\{\Delta l_t = 1\})(1 - 2P\{\Delta l_{t+11} = 1\}) \\ & (1 - 2P\{\Delta(l_{t+6}l_{t+8}) = 1\})(1 - 2P\{\Delta(l_{t+10}l_{t+15}) = 1\}), \end{aligned} \tag{6}$$

where

$$P\{\Delta(l_{t+6}l_{t+8}) = 1\} = (P\{\Delta l_{t+6} = 1\} + P\{\Delta l_{t+8} = 1\} - P\{\Delta l_{t+6} = 1\}P\{\Delta l_{t+8} = 1\}) \cdot \frac{1}{2},$$

$$P\{\Delta(l_{t+10}l_{t+15}) = 1\} = (P\{\Delta l_{t+10} = 1\} + P\{\Delta l_{t+15} = 1\} - P\{\Delta l_{t+10} = 1\}P\{\Delta l_{t+15} = 1\}) \cdot \frac{1}{2}.$$

From Eq. (2), we can obtain the formula to calculate the probability of $\Delta l_{t+19} = 1$:

$$\begin{aligned} &P\{\Delta l_{t+19} = 1\} \\ &= P\{\Delta(s_t \oplus s_{t+5} \oplus s_{t+4}s_{t+7} \oplus s_{t+9}a_t) = 1\} \\ &= \frac{1}{2} - \frac{1}{2} \cdot (1 - 2P\{\Delta s_t = 1\})(1 - 2P\{\Delta s_{t+5} = 1\}) \\ &(1 - 2P\{\Delta(s_{t+4}s_{t+7}) = 1\})(1 - 2P\{\Delta(s_{t+9}a_t) = 1\}), \end{aligned} \tag{7}$$

where

$$P\{\Delta(s_{t+4}s_{t+7}) = 1\} = (P\{\Delta s_{t+4} = 1\} + P\{\Delta s_{t+7} = 1\} - P\{\Delta s_{t+4} = 1\}P\{\Delta s_{t+7} = 1\}) \cdot \frac{1}{2},$$

$$P\{\Delta(s_{t+9}a_t) = 1\} = a_t P\{\Delta s_{t+9} = 1\}.$$

Using the two formulas, we can calculate the probabilities of the differences in the update bits in Algorithm 2 at every subsequent round after the conditions stop being applied. After a certain round the probability forever becomes 1/2. Before that, we can find the biased bit corresponding to the longest conditional differential characteristic.

4 Application to KATAN32

We have applied MILP method to KATAN32 for different rounds to obtain different differential characteristics and minimum conditions. We chose two results with fewer conditions in the previous rounds.

For 64-round KATAN32 (we have modeled 64-round KATAN32 together), the minimum number of conditions is 27. We cannot, however, apply all these conditions, since there are too many key bits and plaintext bits involved in them, resulting in attack failure. We only choose 11 conditions from the first 23 rounds to impose in this analysis. Since other conditions from round 24 have not been applied, difference propagation becomes out of control, with more and more probabilities of differences in update bits tending to be 1/2. We calculate the probabilities of $\Delta s_{t+13} = 1$ and $\Delta l_{t+19} = 1$ after round 23, and we find that finally the probability of $\Delta s_{t+13} = 1$ would always be 1/2 starting from s_{79} and the probability of $\Delta l_{t+19} = 1$ would always be 1/2 starting from l_{82}. Before l_{82}, we detect an obvious bias in Δl_{79}. l_{79} is generated at round 60 and is the

Algorithm 2: Calculating the probabilities of the differences in the update bits from round t to round u

Input: $\{P\{\Delta s_t = 1\}, P\{\Delta s_{t+1} = 1\}, \ldots P\{\Delta s_{t+12} = 1\}\}$: the set of probabilities of the difference for each bit of the 13-bit NLFSR at round t ; $\{P\{\Delta l_t = 1\}, P\{\Delta l_{t+1} = 1\}, \ldots P\{\Delta l_{t+18} = 1\}\}$: the set of probabilities of the difference for each bit of the 19-bit NLFSR at round t.

Output: A : the set of the probabilities of the differences for update bits from round t to round u, there are two update bits at each round.

$S := \{P\{\Delta s_t = 1\}, P\{\Delta s_{t+1} = 1\}, \ldots P\{\Delta s_{t+12} = 1\}\}$;
$L := \{P\{\Delta l_t = 1\}, P\{\Delta l_{t+2} = 1\}, \ldots P\{\Delta s_{t+18} = 1\}\}$;
$A := \emptyset$;
for $i \in \{t, t+1, \ldots, u\}$ **do**
  $P\{\Delta s_{i+13} = 1\}$:= The probability calculated from L according to formula 6 ;
  $P\{\Delta l_{i+19} = 1\}$:= The probability calculated from S according to formula 7 ;
  $S := \{P\{\Delta s_{i+1} = 1\}, P\{\Delta s_{i+2} = 1\}, \ldots P\{\Delta s_{i+13} = 1\}\}$;
  $L := \{P\{\Delta l_{i+1} = 1\}, P\{\Delta l_{i+2} = 1\}, \ldots P\{\Delta l_{i+19} = 1\}\}$;
  $A := A \cup \{(P\{\Delta s_{i+13} = 1\}, P\{\Delta l_{i+19} = 1\})\}$
end
return A;

rightmost bit of the 19-bit NLFSR at round 79. Utilizing the bias of Δl_{79}, we can recover 10 equivalent key bits of the 79-round KATAN32.

For 77-round KATAN32, the minimum number of conditions is 34. We only impose seven conditions from the first 16 rounds and recover four equivalent key bits of the 81-round KATAN32 with a bias in Δl_{81}. l_{81} is generated at round 62 and is the rightmost bit of the 19-bit NLFSR at round 81.

In this section, we present the details of our analysis and attacks on these two results.

4.1 Key-Recovery Attack on 79-Round KATAN32

The differential characteristic of 64-round KATAN32 has the initial difference of weight six at the positions $0, 11, 21, 26, 30, 31$ of the plaintext block, $\Delta X = 0xc4200801$. We only apply 11 conditions in the first 23 rounds.

At round 1, we have $\Delta s_{14} = x_9, \Delta l_{20} = x_{23}$, and we impose conditions $x_9 = 0, x_{23} = 0$. At round 3, we have $\Delta s_{16} = x_5, \Delta l_{22} = x_{24}$, and we impose conditions $x_5 = 0, x_{24} = 0$. At round 6, we have $\Delta l_{25} = s_{13}$, and we impose the condition

$$s_{13} = x_{18} \oplus x_7 \oplus x_{12}x_{10} \oplus x_8x_3 \oplus k_1 = 0. \tag{8}$$

At round 8, we have $\Delta s_{21} = l_{23}$, and we impose the condition

$$l_{23} = x_{27} \oplus x_{22} \oplus k_8 = 0. \tag{9}$$

At round 10, we have $\Delta s_{23} = x_2$, and we impose the condition $x_2 = 0$. At round 12, we have $\Delta s_{25} = l_{20}$, and we impose the condition

$$l_{20} = x_{30} \oplus x_{25} \oplus x_{21} \oplus k_2 = 0. \tag{10}$$

At round 14, we have $\Delta s_{27} = l_{24}$, and we impose the condition

$$l_{24} = x_{26} \oplus x_{21} \oplus x_{22}x_{19} \oplus x_{17} \oplus x_6 \oplus k_3 \oplus k_{10} = 0. \tag{11}$$

At round 19, we have $\Delta s_{32} = l_{34}$. If we try to impose the condition $l_{34} = 0$, it has too many variables, which would make the attack unavailable because of the significantly high computing complexity. So we skip this condition, and assume $P\{\Delta s_{32} = 1\} = 1/2$. At round 21, we have $\Delta s_{34} = l_{27}$, and we impose the condition

$$l_{27} = x_{19}(x_{16} \oplus x_{10}x_8 \oplus x_6x_1 \oplus k_5) \oplus k_{16} = 0. \tag{12}$$

At round 23, we have $\Delta s_{36} = l_{31}$, and we impose the condition

$$\begin{aligned} l_{31} = {} & x_{19} \oplus x_{14} \oplus x_3 \oplus x_8x_6 \oplus x_4(x_{31} \oplus x_{26} \oplus x_{22} \oplus k_0) \oplus \\ & (x_{15} \oplus x_4 \oplus k_7)(x_{12} \oplus x_1 \oplus x_6x_4 \oplus k_{13}) \oplus k_9 \oplus k_{24} = 0. \end{aligned} \tag{13}$$

The difference propagation and the conditions applied are presented in Table 8.

After imposing these conditions, we obtain the probability of difference in each bit at round 24 as follows: $(0, 0, 0, 0, 0, 0, 0, \frac{1}{2}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

According to Algorithm 2, we can compute the bias of the difference in the update bit for each round after round 24, and find that starting from l_{82} the probability of $\Delta l_{t+19} = 1$ would always be $1/2$, and among the bits whose positions are very close to l_{82}, l_{79} has the maximum biased difference, shown as follows:

$$P\{\Delta l_{79} = 1 | \text{all the conditions satisfied}\} \approx 0.5 - 0.00001.$$

We confirmed the strongly biased difference in bit l_{79} experimentally. Let us consider the conditions applied. There are 10 equivalent key bits k_0, k_1, k_2, $k_3 \oplus k_{10}$, k_5, k_7, k_8, $k_9 \oplus k_{24}$, k_{13}, k_{16} and 21 bits of plaintext x_1, x_3, x_4, x_6, x_7, x_8, x_{10}, x_{12}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{21}, x_{22}, x_{25}, x_{26}, x_{27},x_{30},x_{31} involved in the conditions. We choose 2^8 key in which bits $k_0, k_1, k_2, k_3, k_5, k_7, k_8, k_9$ are free and others fixed. For each key, we enumerate 2^{21} plaintexts of which the 21 bits involved in the conditions are free and other bits are zero. We then can use the conditions (8)–(13) to filter the 2^{21} plaintexts, and if the plaintext satisfied the conditions, we calculate Δl_{79} with the initial difference $\Delta X = 0xc4200801$ and count $P\{\Delta l_{79} = 1\}$ at last. The complexity of each experiment is less than $2^{21+1} = 2^{22}$ evaluations of the 60-round KATAN32 encryption while not every plaintext can be filtered. The experimental results verify the strongly biased difference in bit l_{79}. All the results of these 256 experiments are that $P\{\Delta l_{79} = 1\}$ is lower than $0.5 - 0.00001$.

Furthermore, we can mount a key-recovery attack. Looking at the conditions (8)–(13), we consider k_0, k_1, k_2, $k_3 \oplus k_{10}$, k_5, k_7, k_8, $k_9 \oplus k_{24}$, k_{13}, k_{16} , the 10 equivalent key bits, as ten variables. In a key-recovery attack, since the key is unknown to the attacker, we enumerate 2^{10} guesses of these 10 equivalent key bits. For each guess, similar to the verification, we use conditions (8)–(13) to filter 2^{21} plaintexts of which the 21 bits involved in the conditions (8)–(13) are free and other 11 bits are fixed to zero, then calculate Δl_{79} with initial difference $\Delta X = 0xc4200801$ and finally count $P\{\Delta l_{79} = 1\}$.

When the guess is correct, plaintexts are filtered by the conditions corresponding to the correct guessed equivalent key bits, then $P\{\Delta l_{79} = 1\}$ shows the obvious bias. In the 1024 statistical results from 1024 guesses of 10 equivalent key bits, the maximum bias in the results corresponds to the correct values of the ten equivalent key bits. This allows us to recover k_0, k_1, k_2, $k_3 \oplus k_{10}$, k_5, k_7, k_8, $k_9 \oplus k_{24}$, k_{13}, k_{16}, with experimental complexity less than $2^{10+21+1} = 2^{32}$ evaluations of the 60-round KATAN32 encryption. We randomly chose four 80-bit keys and mounted four key-recovery attack experiments, and each time the 10 equivalent key bits were recovered correctly, as shown by the results listed in Table 6.

Table 6. Four key-recovery attack experiments on 79-round KATAN32

No.	80-bit key	Equivalent key bits with the maximum bias									
		k_0	k_1	k_2	$k_3 \oplus k_{10}$	k_5	k_7	k_8	$k_9 \oplus k_{24}$	k_{13}	k_{16}
1	$0x68b1644ead28b1644e8e$	0	1	1	1	0	0	1	0	0	0
2	$0xf8b164cead28b1644e8e$	1	1	1	0	0	0	1	1	0	0
3	$0x38116486a99ab3664d9e$	0	0	1	1	0	0	0	1	0	0
4	$0x2d11e4062b92bb2e4d9f$	0	0	1	0	1	1	0	0	0	1

4.2 Key-Recovery Attack on 81-Round KATAN32

The initial difference of the differential characteristic of 77-round KATAN32 weights three at position 7, 18 and 28 of the plaintext block, $\Delta X = 0x10040080$.

At round 1, we have $\Delta s_{14} = x_2$, then impose the condition $x_2 = 0$ to prevent difference propagation.

Similarly, at round 3, 5, 7, we have $\Delta s_{16} = x_9, \Delta s_{18} = x_5, \Delta s_{20} = x_1$, so we require bits x_9, x_5, x_1 to be zero.

At round 12, we have $\Delta s_{25} = l_{27}$, and we impose the condition

$$\begin{aligned} l_{27} = {} & x_{23} \oplus x_{18} \oplus x_7 \oplus x_{12}x_{10} \oplus x_8x_3 \oplus x_{19}(x_{16} \\ & \oplus x_{10}x_8 \oplus k_5) \oplus k_{16} \oplus k_1 = 0. \end{aligned} \tag{14}$$

At round 14, we have $\Delta s_{27} = l_{20}$ and we impose the condition

$$l_{20} = x_{30} \oplus x_{25} \oplus x_{26}x_{23} \oplus x_{21} \oplus k_2 = 0. \tag{15}$$

At round 16, we have $\Delta s_{29} = l_{24}$ and we impose the condition

$$l_{24} = x_{26} \oplus x_{21} \oplus x_{22}x_{19} \oplus x_{17} \oplus x_6 \oplus k_3 \oplus k_{10} = 0. \tag{16}$$

The differences in propagation and conditions applied are presented in Table 9. After imposing these conditions, we obtain the probability of difference in each bit at round 17 as follows: (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0).

We compute the bias of the update bit of each round from the 17th round and find that starting from l_{84} the probability of $\Delta l_{t+19} = 1$ would always be 1/2. Among the bits whose positions are very close to l_{84}, l_{81} has the maximum biased difference, shown as follows:

$$P\{\Delta l_{81} = 1 | \text{all the conditions satisfied}\} \approx 0.5 + 0.000226.$$

We experimentally verified the strongly-biased difference in bit l_{81} . There are four equivalent key bits $k_5, k_{16} \oplus k_1, k_2, k_3 \oplus k_{10}$ and 16 bits of plaintext x_{30}, x_{26}, x_{25}, x_{23}, x_{22}, x_{21}, x_{19}, x_{18}, x_{17}, x_{16}, x_{12}, x_{10}, x_8, x_7, x_6, x_3 in conditions (14)–(16). We choose 2^4 keys of which k_1, k_2, k_3, k_5 are free and the others fixed. For each key, we enumerate 2^{16} plaintexts of which the 16 bits involved in conditions are free and other bits are fixed to 0. We then use conditions (14)–(16) to filter the 2^{16} plaintexts, and if a plaintext satisfies the conditions, we calculate Δl_{81} with the initial difference $\Delta x = 0x10040080$ and count $P\{\Delta l_{81} = 1\}$. The complexity of each experiment is less than $2^{16+1} = 2^{17}$ evaluations of the 62-round KATAN32 encryption. In all the results of these 16 experiments $P\{\Delta l_{81} = 1\}$ is greater than $0.5 + 0.000226$.

We now will describe mounting the key-recovery attack. Looking at conditions (14)–(16), we consider $k_1 \oplus k_{16}, k_2, k_3 \oplus k_{10}, k_5$ these four equivalent key bits as four boolean variables. There are 16 bits of plaintext involved in conditions (14)–(16). To enlarge the space of plaintexts after filtering, we choose other three bits of plaintext not included in any condition as free bits in addition to the 16 bits of plaintext involved in conditions (14)–(16). For each of the 2^4 guesses of these four variables, we use conditions (14)–(16) to filter the 2^{19} plaintexts enumerated by the 19 bits we just choose with the remaining 13 bits fixed to 0. We then calculate Δl_{81} with initial difference $\Delta x = 0x10040080$ and count $P\{\Delta l_{81} = 1\}$. In the 16 statistical results obtained from 16 guesses of four equivalent key bits, the maximum bias in the results corresponds to the correct value of the four equivalent key bits, allowing us to recover $k_1 \oplus k_{16}, k_2, k_3 \oplus k_{10}, k_5$. The complexity of the experiment is less than $2^{4+19+1} = 2^{24}$ evaluations of the 62-round KATAN32 encryption. We chose five 80-bit keys randomly and mounted five key-recovery attack experiments, and each time the four equivalent key bits can be correctly recovered. The results of these five key-recovery attack experiments are listed in Table 7.

Table 7. Five key-recovery attack experiments on 81-round KATAN32

No.	80-bit key	Equivalent key bits with the maximum bias			
		$k_1 \oplus k_{16}$	k_2	$k_3 \oplus k_{10}$	k_5
1	0x68b1644ead28b1644e8e	1	1	1	0
2	0x48b1644ead28b1644e8e	1	0	1	0
3	0xcda964ceb98cb7644e8e	1	0	1	1
4	0xc9a1448cb886b7644f86	1	0	1	0
5	0x99a1448cb88680644f86	0	0	0	0

5 Extension with the Standard Differential Attack

Combined with the standard differential attack, the conditional differential attack on 81-round KATAN32 can be extended to 97-round, 98-round and 99-round key-recovery attacks.

5.1 Key-Recovery Attack on 97-Round KATAN32

Inspired by the technique representing the dependence of the intermediate state on the output by an algebraic representation in [27], we give the algebraic representation of the intermediate state using the ciphertext and round keys.

Using Eq. (1) and Eq. (2), we can get the expression of l_t, s_t in decryption direction:

$$l_t = s_{t+13} \oplus l_{t+11} \oplus l_{t+6}l_{t+8} \oplus l_{t+10}l_{t+15} \oplus k_{2t+1}, \tag{17}$$

$$s_t = l_{t+19} \oplus s_{t+5} \oplus s_{t+4}s_{t+7} \oplus s_{t+9}a_t \oplus k_{2t}, \tag{18}$$

Suppose the output bits of 97-round KATAN32 corresponding to plaintext X are $S_{97} = (s_{97}, s_{98}, \ldots, s_{110})$ and $L_{97} = (l_{97}, l_{98}, \ldots, l_{115})$, and the output bits of 97-round KATAN32 corresponding to plaintext $X + \Delta X$ are $S'_{97} = (s'_{97}, s'_{98}, \ldots, s'_{110})$ and $L'_{97} = (l'_{97}, l'_{98}, \ldots, l'_{115})$. For decryption direction, Δl_{81} can be expressed by round keys and the ciphertext of 97-round KATAN32 by using Eq. (17) and Eq. (18) iteratively.

$\Delta l_{81} = l_{113} \oplus s_{99} \oplus s_{98}s_{101} \oplus s_{105} \oplus l_{103} \oplus l_{98}l_{100} \oplus l_{102}l_{107} \oplus (s_{100} \oplus l_{98} \oplus (s_{106} \oplus l_{104} \oplus l_{99}l_{101} \oplus l_{103}l_{108} \oplus k_{187})(s_{108} \oplus l_{106} \oplus l_{101}l_{103} \oplus l_{105}l_{110} \oplus k_{191}) \oplus l_{97}l_{102} \oplus k_{175})(s_{102} \oplus l_{100} \oplus (s_{108} \oplus l_{106} \oplus l_{101}l_{103} \oplus l_{105}l_{110} \oplus k_{191})l_{97} \oplus l_{99}l_{104} \oplus k_{179}) \oplus (s_{104} \oplus l_{102} \oplus l_{97}l_{99} \oplus l_{101}l_{106} \oplus k_{183})(s_{109} \oplus l_{107} \oplus l_{102}l_{104} \oplus l_{106}l_{111} \oplus k_{193}) \oplus l'_{113} \oplus s'_{99} \oplus s'_{98}s'_{101} \oplus s'_{105} \oplus l'_{103} \oplus l'_{98}l'_{100} \oplus l'_{102}l'_{107} \oplus (s'_{100} \oplus l'_{98} \oplus (s'_{106} \oplus l'_{104} \oplus l'_{99}l'_{101} \oplus l'_{103}l'_{108} \oplus k_{187})(s'_{108} \oplus l'_{106} \oplus l'_{101}l'_{103} \oplus l'_{105}l'_{110} \oplus k_{191}) \oplus l'_{97}l'_{102} \oplus k_{175})(s'_{102} \oplus l'_{100} \oplus (s'_{108} \oplus l'_{106} \oplus l'_{101}l'_{103} \oplus l'_{105}l'_{110} \oplus k_{191})l'_{97} \oplus l'_{99}l'_{104} \oplus k_{179}) \oplus (s'_{104} \oplus l'_{102} \oplus l'_{97}l'_{99} \oplus l'_{101}l'_{106} \oplus k_{183})(s'_{109} \oplus l'_{107} \oplus l'_{102}l'_{104} \oplus l'_{106}l'_{111} \oplus k_{193})$.

According to this expression, one can calculate Δl_{81} by using the ciphertexts of 97-round KATAN32 and 6 equivalent key bits $k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}$. We extend the attack described in Subsect. 4.2 to 97-round as follows. Plaintexts being filtered by the conditions are encrypted to get ciphertexts by 97-round

Table 8. Differential characteristic and conditions for $\Delta X = 0xc4200801$

Round	Difference state	conditions
0	1100010000100 0000000100000000001	
1	1000100001000 0000001000000000010	$x_9 = 0, x_{23} = 0$
2	0001000010000 0000010000000000100	
3	0010000100000 0000100000000001000	$x_5 = 0, x_{24} = 0$
4	0100001000000 0001000000000010000	
5	1000010000000 0010000000000100000	
6	0000100000000 0100000000001000000	condition (8)
7	0001000000000 1000000000010000000	
8	0010000000000 0000000001000000000	condition (9)
9	0100000000000 0000000010000000000	
10	1000000000000 0000000100000000000	$x_2 = 0$
11	0000000000000 0000001000000000001	
12	0000000000000 0000010000000000010	condition (10)
13	0000000000000 0000100000000000100	
14	0000000000000 0001000000000001000	condition (11)
15	0000000000000 0010000000000010000	
16	0000000000000 0100000000000100000	
17	0000000000000 1000000000001000000	
18	0000000000000 0000000000010000000	
19	0000000000000 0000000000100000000	
20	000000000000* 0000000001000000000	
21	00000000000*0 0000000010000000000	condition(12)
22	0000000000*00 0000000100000000000	
23	000000000*000 0000001000000000000	condition(13)
24	00000000*0000 0000010000000000000	

The red bits denote the update bits.
The blue bits denote the bits that generate the update bits.
The differential probability of the bit $*$ is $\frac{1}{2}$

KATAN32. Δl_{81} could be computed from ciphertexts of 97-round KATAN32 and the guess of these 6 equivalent key bits $k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}$. Given every guess of 10 equivalent key bits ($k_1 \oplus k_{16}$, k_2, $k_3 \oplus k_{10}$, k_5, k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}), we can calculate and count Δl_{81} with respect to a set of filtered plaintexts. If the guess is right, the $P\{\Delta l_{81} = 1\}$ shows a obvious bias. The computational cost of the experiment is less than 2^{24+6} encryptions of 97-round KATAN32. We mounted five key-recovery attack experiments with the same key as the experiments in Subsect. 4.2 and each time the 10 equivalent key bits can be correctly recovered.

Table 9. Differential characteristic and conditions for $\Delta X = 0x10040080$

Round	Difference state	conditions
0	0001000000000 10000000000010000000	
1	0010000000000 00000000001000000000	$x_2 = 0$
2	0100000000000 00000000010000000000	
3	1000000000000 00000000100000000000	$x_9 = 0$
4	0000000000000 00000001000000000001	
5	0000000000000 00000010000000000010	$x_5 = 0$
6	0000000000000 00000100000000000100	
7	0000000000000 00001000000000001000	$x_1 = 0$
8	0000000000000 00010000000000010000	
9	0000000000000 00100000000000100000	
10	0000000000000 01000000000001000000	
11	0000000000000 10000000000010000000	
12	0000000000000 00000000001000000000	condition(14)
13	0000000000000 00000000010000000000	
14	0000000000000 00000000100000000000	condition(15)
15	0000000000000 00000001000000000000	
16	0000000000000 00000010000000000000	condition(16)
17	0000000000000 00000100000000000000	

The red bits denote the update bits.
The blue bits denote the bits that generate the update bits.

5.2 Key-Recovery Attack on 98-Round KATAN32

If Δl_{81} is expressed by the ciphertext of 98-round KATAN32 and round keys, there are 7 equivalent key bits involved in the expression. So the computational cost of the key-recovery attack is less than 2^{24+7} encryptions of 98-round KATAN32. In this attack, 11 equivalent key bits $k_1 \oplus k_{16}$, k_2, $k_3 \oplus k_{10}$, k_5, k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}, k_{195} can be correctly recovered. Every experiment requires about 2.4 hours on a 2.5 Ghz PC with our implementation.

5.3 Key-Recovery Attack on 99-Round KATAN32

If Δl_{81} is expressed by the ciphertext of 99-round KATAN32 and round keys, there are 9 equivalent key bits involved in the expression. So the computational cost of the key-recovery attack is less than 2^{24+9} encryptions of 99-round KATAN32. In this attack, 13 equivalent key bits $k_1 \oplus k_{16}$, k_2, $k_3 \oplus k_{10}$, k_5, k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}, k_{195}, k_{196}, k_{197} can be correctly recovered. Every experiment requires about 9.64 hours on a 2.5 Ghz PC with our implementation.

It is thus possible to extend the conditional differential attack on 81-round KATAN32 to 114-round with a time complexity of 2^{63} encryptions.

6 Conclusion

In this paper, we propose two strategies for improving conditional differential analysis on the NLFSR based block cipher KATAN32. We first apply the MILP model to automatically search for the conditional differential characteristic of NLFSR based block ciphers, helping us efficiently obtain the initial difference and conditions of the conditional differential analysis. We propose a new method to calculate the probability of the difference to help quickly detect the bit with a bias. We apply the improved conditional differential analysis to KATAN32 and obtain two results, recovering 10 equivalent key bits of 79-round KATAN32 and four equivalent key bits of 81-round KATAN32, respectively.

Combined with standard differential attack, we extend the 81-round conditional key-recovery attack to 99-round with the time complexity being 2^{33} encryptions of 99-round KATAN32 and recover 13 equivalent key bits. Compared with previously best practical distinguisher on KATAN32, our results are extended more than 7 rounds with less cost of computation time and memory. We believe both strategies to be general to NLFSR based ciphers. Applying these two strategies on other NLFSR based ciphers will be one topic of interest in our future works.

Acknowledgements. We are very grateful to the anonymous reviewers. This work was supported by the National Natural Science Foundation of China under Grant 61672330, and Grant 11771256.

References

1. Ahmadian, Z., Rasoolzadeh, S., Salmasizadeh, M., Aref, M.R.: Automated dynamic cube attack on block ciphers: cryptanalysis of SIMON and KATAN. IACR Cryptology ePrint Archive 2015, 40 (2015). http://eprint.iacr.org/2015/040
2. Albrecht, M.R., Leander, G.: An all-in-one approach to differential cryptanalysis for small block ciphers. IACR Cryptology ePrint Archive 2012, 401 (2012). http://eprint.iacr.org/2012/401
3. Ben-Aroya, I., Biham, E.: Differential cryptanalysis of lucifer. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 187–199. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_17
4. Biham, E., Dunkelman, O.: Differential cryptanalysis in stream ciphers. IACR Cryptology ePrint Archive 2007, 218 (2007). http://eprint.iacr.org/2007/218
5. Bogdanov, A., et al.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74735-2_31
6. 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). https://doi.org/10.1007/978-3-642-04138-9_20
7. De Cannière, C., Küçük, Ö., Preneel, B.: Analysis of grain's initialization algorithm. In: Vaudenay, S. (ed.) AFRICACRYPT 2008. LNCS, vol. 5023, pp. 276–289. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68164-9_19

8. Abed, F., et al.: Pipelineable on-line encryption. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 205–223. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46706-0_11
9. Fuhr, T., Minaud, B.: Match box meet-in-the-middle attack against KATAN. In: Cid and Rechberger [8], pp. 61–81. https://doi.org/10.1007/978-3-662-46706-0_4
10. Huang, S., Wang, X., Xu, G., Wang, M., Zhao, J.: Conditional cube attack on reduced-round Keccak sponge function. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 259–288. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_9
11. Isobe, T.: A single-key attack on the full GOST block cipher. J. Cryptol. **26**(1), 172–189 (2013). https://doi.org/10.1007/s00145-012-9118-5
12. Isobe, T., Shibutani, K.: Improved all-subkeys recovery attacks on fox, KATAN and SHACAL-2 block ciphers. In: Cid and Rechberger [8], pp. 104–126. https://doi.org/10.1007/978-3-662-46706-0_6
13. Jiang, Z., Jin, C.: Impossible differential cryptanalysis of 8-round Deoxys-BC-256. IEEE Access **6**, 8890–8895 (2018). https://doi.org/10.1109/ACCESS.2018.2808484
14. 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). https://doi.org/10.1007/978-3-642-17373-8_8
15. 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). https://doi.org/10.1007/978-3-642-28496-0_12
16. Li, Z., Bi, W., Dong, X., Wang, X.: Improved conditional cube attacks on Keccak keyed modes with MILP method. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 99–127. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_4
17. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu, C.-K., Yung, M., Lin, D. (eds.) Inscrypt 2011. LNCS, vol. 7537, pp. 57–76. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34704-7_5
18. Rasoolzadeh, S., Raddum, H.: Multidimensional meet in the middle cryptanalysis of KATAN. IACR Cryptology ePrint Archive 2016, 77 (2016). http://eprint.iacr.org/2016/077
19. Sasaki, Yu., Todo, Y.: New impossible differential search tool from design and cryptanalysis aspects. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10212, pp. 185–215. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_7
20. Song, L., Guo, J., Shi, D.: New MILP modeling: improved conditional cube attacks to Keccak-based constructions. IACR Cryptology ePrint Archive 2017, 1030 (2017). http://eprint.iacr.org/2017/1030
21. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 158–178. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_9
22. Wang, D., Li, W., Wang, P.: Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks. IEEE Trans. Ind. Inform. **14**(9), 4081–4092 (2018). https://doi.org/10.1109/TII.2018.2834351

23. Wei, L., Rechberger, C., Guo, J., Wu, H., Wang, H., Ling, S.: Improved meet-in-the-middle cryptanalysis of KTANTAN. IACR cryptology eprint archive 2011, 201 (2011)
24. Wu, H., Preneel, B.: Resynchronization attacks on WG and LEX. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 422–432. Springer, Heidelberg (2006). https://doi.org/10.1007/11799313_27
25. Wu, S., Wang, M.: Security evaluation against differential cryptanalysis for block cipher structures. IACR Cryptology ePrint Archive 2011, 551 (2011). http://eprint.iacr.org/2011/551
26. Xiang, Z., Zhang, W., Bao, Z., Lin, D.: Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 648–678. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_24
27. Zhang, W., Cao, M., Guo, J., Pasalic, E.: Improved security evaluation of SPN block ciphers and its applications in the single-key attack on SKINNY. IACR Trans. Symmetric Cryptol. **2019**(4), 171–191 (2019). https://doi.org/10.13154/tosc.v2019.i4.171-191
28. Zhu, B., Gong, G.: Multidimensional meet-in-the-middle attack and its applications to KATAN32/48/64. Cryptog. Commun. **6**(4), 313–333 (2014). https://doi.org/10.1007/s12095-014-0102-9