Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8883557, 14 pages
https://doi.org/10.1155/2020/8883557

Research Article

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*
[2]*School of Science, Shandong Jiaotong University, Jinan 250357, China*
[3]*School of Management Engineering, Shandong Jianzhu University, Jinan 250101, China*

Correspondence should be addressed to Wenying Zhang; zhangwenying@sdnu.edu.cn

Received 24 June 2020; Revised 18 October 2020; Accepted 3 November 2020; Published 23 November 2020

Academic Editor: Ding Wang

In this paper, a new method for constructing a Mixed Integer Linear Programming (MILP) model on conditional differential cryptanalysis of the nonlinear feedback shift register- (NLFSR-) based block ciphers is proposed, and an approach to detecting the bit with a strongly biased difference is provided. 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 and 13 equivalent key bits of 99-round KATAN32. The time complexity is less than 2^{31} encryptions of 98-round KATAN32 and less than 2^{33} encryptions of 99-round KATAN32, respectively. Thus far, our results are the best known practical key-recovery attacks for the round-reduced variants of KATAN32 regarding the number of rounds and the time complexity. All the results are verified experimentally.

1. Introduction

Cryptographic techniques move into applications like access control, parking management, goods tracking, radio frequency identification tags, and integrated circuit (IC) printing [1]. At the same time, wireless sensor networks (WSNs) have been used for various critical industrial applications, such as heartbeat monitoring, temperature monitoring for precision agriculture, self-monitoring of autonomous vehicles, and power usage monitoring for smart grid [2, 3]. In these new cryptography environments, RFID technology applications and sensor networks have similar features such as weak computation ability, small storage space, and strict power constraints. However, the data processed in these applications are sensitive [4]. The ever-increasing demand for security and privacy in these very constrained environments requires new cryptographic primitives, like low cost, tiny, and efficient ciphers. Hence, traditional block ciphers such as AES are not suitable for these constrained environments. Many lightweight ciphers, including KATAN and KTANTAN family [5] and Piccolo [6], have been proposed to tackle this problem.

The KATAN and KTANTAN block ciphers were proposed by Christophe DeCannière, Orr Dunkelman, and Miroslav Knezevic at CHES 2009 [5]. In order to reduce the energy consumed in data processing and improve the efficiency, KATAN uses nonlinear feedback shift registers (NLFSRs) as well as a linear key schedule [7]. 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 $K = k_{79}k_{78}\cdots k_1k_0$ are selected each round. However, 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). For each round, positions 0 and 1 of the LFSR are generated as the round subkey

Table 1: Cryptanalytic results on KATAN32.

Type	Scenario	Rounds	Time complexity	Reference
Conditional differential	Single key	78	2^{22}	[9]
	Related key	120	2^{31}	[11]
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*}	[12]
	Single key	115	2^{78}	
MIMT ASR	Single key	119	$2^{79.1}$	[13]
Match Box MITM	Single key	153	$2^{78.5}$	[14]
Dynamic Cube	Single key	155	$2^{78.3}$	[15]
Multidimensional MITM	Single key	175	$2^{79.3}$	[16]
	Single key	206	2^{79}	[17]

*A 91-round differential distinguisher.

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. [8], 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 and KTANTAN [9] using conditional differential cryptanalysis [10] 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 and then obtained key-recovery attacks for 120 of 254 rounds of KATAN32 [11]. Finding the nonuniformity of the difference distribution after 91 rounds, Albrecht and Leander proposed a 91-round distinguisher with the time complexity being 2^{32} encryptions [12]. These results on KATAN32 are listed in Table 1.

Other types of attacks formally published on this cipher are also listed in Table 1, such as all subkeys recovery (ASR), which is a variant of the meet-in-the-middle (MITM) attack [13], Match Box MITM attack [14], Dynamic Cube attack [15], and Multidimensional MITM attack [16, 17]. As can be seen from the details in Table 1, each time complexity is too high to present a practical attack.

As stated in [18], 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 as a passive RFID tag may not be changed during its life cycle. In a practical sense, the security of a lightweight cipher under the single-key scenario is the most important. As shown in [19], 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 in [10]. The idea is to control the propagation of differences by imposing conditions on the public variables of the cipher. In particular, we want to impose some conditions to filter plaintexts. Depending on whether these conditions involve secret variables or not, key-recovery or distinguishing attacks can be mounted. The key bit conditions lead to a key-recovery attack. The technique has been extended to higher order differential cryptanalysis. Later, it has been a very popular technique in hash functions cryptanalysis [20]. It allows increasing the probability of a differential characteristic satisfying some conditions; it also can be useful for block ciphers.

In some attacks, attackers derive the conditions by hand, which is time consuming and error prone. This paper uses an automatic tool named Mixed Integer Linear Programming (MILP) to get minimum conditions and obtain new cryptanalytic results. 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 [21] and Wu et al. in [22] to count the active Sboxes of word-based block ciphers. It has been applied to search for differential characteristics and linear approximations [23, 24]. It has also been applied to search for integral distinguishers and division trails [25, 26] and impossible differentials [27, 28]. 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 [29–31].

1.2. Our Contributions. In this paper, we improve 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 the fewer the conditions, the higher the probability of the differential path. On the other hand, we propose a method to quickly calculate the bias of every bit quickly and detect the bit, which has a strongly biased difference. Finally, using the standard differential attack, we extend 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 [9], Knellwolf et al. 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 by 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 of rounds. Based on the method using MILP, we automatically obtain the initial difference and conditions.

Second, we present an approach to detecting the bias in the difference of the update bit. In [9], Knellwolf et al. detected the bias experimentally by observing certain nonrandomness 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 bit's position with a bias, we can 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}, and 2^{33} encryptions, respectively. Extended key-recovery attacks can recover 10, 11, and 13 equivalent key bits, respectively. It is the best known practical cryptanalytic result on KATAN32 so far.

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

1.3. Organization. The paper is organized as follows. In Section 2, some preliminaries are introduced. Section 3 describes the two improvements in conditional differential attacks. In Section 4, with these improvements, the attacks mounted on 79 and 81 of 254 rounds of KATAN32 are presented in detail. In Section 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 Section 6.

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. Preliminaries

We present our notations in Table 2.

2.1. Description of KATAN. The block ciphers KATAN family are lightweight cryptographic primitives dedicated to hardware implementation. They share a very similar structure based on nonlinear feedback shift registers (NLFSR). KATAN and KTANTAN are composed of three block ciphers with 32-, 48-, and 64-bit block sizes, respectively, denoted by KATANn and KTANTANn for $n = 32,48,64$. They all have 80-bit keys, and the only difference between KATAN and KTANTAN is the key schedule. The round key bits of KATAN are the linear combination of the initial key bits, and the key bits of KTANTAN are extracted directly from the initial 80 key bits according to the predefined rule. Here, we will briefly introduce KATAN32, which is analyzed in this paper.

2.1.1. 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:

$$k_{i+80} = k_i \oplus k_{i+19} \oplus k_{i+30} \oplus k_{i+67}, \quad 0 \le i \le 427. \tag{1}$$

In the remainder of this paper, for any $i \ge 80$, we call k_i one equivalent key bit, which is the linear combination of the initial key bits.

2.1.2. Round Function. In initialization, a 32-bit plaintext block $X = (x_{31}, \cdots, x_0)$ 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 $S_t = (s_t, s_{t+1}, \cdots, s_{t+12})$ and $L_t = (l_t, l_{t+1}, \cdots, l_{t+18})$.

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

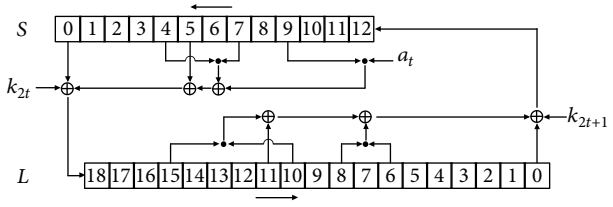

Figure 1: The round function of KATAN32 cipher.

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

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

$$l_{t+19} = s_t \oplus s_{t+5} \oplus s_{t+4}s_{t+7} \oplus s_{t+9}a_t \oplus k_{2t}, \quad (3)$$

where a_t is a round constant generated by the 8-bit LFSR using the recursive relation $a_t = a_{t-3} \oplus a_{t-5} \oplus a_{t-7} \oplus a_{t-8} (t \ge 8)$ 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 Figure 1.

2.2. Conditional Differential Analysis. Knellwolf et al. applied conditional differential cryptanalysis to NLFSR-based cryptosystems at ASIACRYPT 2010 [9]. This technique is based on differential cryptanalysis used to analyze initialization mechanisms of stream ciphers in [32, 33]. After choosing an initial difference, it studies the propagation of the difference through NLFSR-based cryptosystems and identifies conditions on internal state bits to prevent difference propagation whenever possible. By taking the plaintext pairs conforming to 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, there are single key bits or 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 [9], the authors traced differences through cryptosystems and prevented the propagation whenever possible by identifying conditions on internal state variables. They gave suggestions on manually choosing an initial difference rather than providing a specific method for acquiring it. They suggest that the difference propagation should be controllable for as many rounds as possible with fewer conditions. They also suggest 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 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 Section 3.1, one can formulate an MILP model of difference propagation, search for a differential characteristic with minimum conditions, and obtain the initial difference simultaneously.

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

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 NLFSRs can be easily derived when the conditions cease being applied. Taking this probability as the input of the method described in Section 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 the number of rounds is the 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 located update bit will show the bias. The 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, we obtain an initial difference and a conditional differential characteristic with the fewest conditions. The steps are as follows.

(1) Finding All Modes of Difference Propagation under the Control of Conditions. For KATAN32, at each round, only two bits are generated by some bits from the previous round, so the differences in these two bits are caused only by these bits. Equations (2) and (3) show the relation between these bits.

There are linear and nonlinear terms in Equations (2) and (3). If there are differences in nonlinear terms, the difference in the update bit can be canceled by imposing conditions even if there are differences in linear terms at the same time. 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 the difference appears in the update bit.

For example, for Equation (2): $s_{t+13} = l_t \oplus l_{t+11} \oplus l_{t+6}l_{t+8} \oplus l_{t+10}l_{t+15} \oplus k_{2t+1}$, 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 having no differences, no conditions could cancel the difference. The

Input: Equation (2) and Equation (3)
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 bit after round r calculated by λ using the method described in Section 3.2};
$t \leftarrow$ the bit derived from P having the nonzero 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);
  $A \leftarrow A \cup \{(g, |P\{\Delta t = 1\} - 1/2|)\}$;
end
searching in A for the max $|P\{\Delta t = 1\} - (1/2)|$;
return g in accordance with the max $|P\{\Delta t = 1\} - (1/2)|$.

Algorithm 1. The framework of the conditional differential attack.

difference 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. At some other values, there are no conditions that can 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 is aimed at preventing the propagation of differences.

With respect to Equation (2), s_{t+13} is generated by six 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 six 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 $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}, \Delta s_{t+13}, c)$ following the following strategies.

If $\Delta s_{t+13} = 0$ according to Equation (2), c takes value 0.

Example 1. If $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}) = (1, 1, 0, 0, 0, 0)$, according to Equation (2), $\Delta s_{t+13} = 0$. Since no conditions need to be added, $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}, \Delta s_{t+13}, c) = (1, 1, 0, 0, 0, 0, 0, 0)$ is the vector we hunt.

Assuming that Δs_{t+13} may be 1 or 0 according to Equation (2). 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 $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}) = (0, 0, 0, 0, 0, 1)$, according to Equation (2), 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 $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}) = (1, 0, 0, 0, 0, 0)$, according to Equation (2), $\Delta s_{t+13} = 1$, 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 $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15})$ 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 Equation (2), we get all 64 values of the 8-dimensional vector $(\Delta l_t, \Delta l_{t+11}, \Delta l_{t+6}, \Delta l_{t+8}, \Delta l_{t+10}, \Delta l_{t+15}, \Delta s_{t+13}, c)$, presented in Table 3.

Meanwhile, with respect to Equation (3), we can also find all the difference state values $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta s_{t+9}, \Delta l_{t+19}, c)$. It should be noted that in Equation (3) there is a

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)	

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)	

constant a_t at each round. To simplify constraints of the MILP, we model two cases corresponding to the values of a_t.

When $a_t = 1$, Equation (3) contains five Boolean variables s_t, s_{t+5}, s_{t+4}, s_{t+7}, and s_{t+9} so that the difference state $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta s_{t+9})$ can take on one of $2^5 = 32$ different values deriving the 32 values of the 7-dimensional vector $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta s_{t+9}, \Delta l_{t+19}, c)$ shown in Table 4.

When $a_t = 0$, Equation (3) contains four Boolean variables s_t, s_{t+5}, s_{t+4}, and s_{t+7} so that the difference state $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7})$ can take on one of $2^4 = 16$ different values that lead to the 16 values of the 6-dimensional vector $(\Delta s_t, \Delta s_{t+5}, \Delta s_{t+4}, \Delta s_{t+7}, \Delta l_{t+19}, c)$ shown in Table 5.

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)	

(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. This set of linear inequalities characterizes the difference propagation of Equation (2) under the control of conditions. Ten inequalities are remaining after a simple reduction. L_1 shows the ten inequalities.

$$L_1 = \begin{cases} -\Delta s_{t+13} - c + 1 \geq 0, \\ -\Delta l_{t+6} + c \geq 0, \\ -\Delta l_t - \Delta l_{t+11} - \Delta s_{t+13} + 2 \geq 0, \\ -\Delta l_{t+8} + c \geq 0, \\ -\Delta l_{t+10} + c \geq 0, \\ -\Delta l_{t+15} + c \geq 0, \\ \Delta l_t + \Delta l_{t+11} - \Delta s_{t+13} \geq 0, \\ -\Delta l_t + \Delta l_{t+11} + \Delta s_{t+13} + c \geq 0, \\ \Delta l_t - \Delta l_{t+11} + \Delta s_{t+13} + c \geq 0, \\ \Delta l_{t+6} + \Delta l_{t+8} + \Delta l_{t+10} + \Delta l_{t+15} - c \geq 0. \end{cases} \tag{4}$$

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:

$$L_2 = \begin{cases} -\Delta l_{t+19} - c + 1 \geq 0, \\ -\Delta s_{t+7} + c \geq 0, \\ -\Delta s_{t+4} + c \geq 0, \\ \Delta s_{t+4} + \Delta s_{t+7} - c \geq 0, \\ \Delta s_t - \Delta s_{t+5} - \Delta s_{t+9} - \Delta l_{t+19} + 2 \geq 0, \\ -\Delta s_t + \Delta s_{t+5} - \Delta s_{t+9} - \Delta l_{t+19} + 2 \geq 0, \\ -\Delta s_t - \Delta s_{t+5} + \Delta s_{t+9} - \Delta l_{t+19} + 2 \geq 0, \\ \Delta s_t + \Delta s_{t+5} + \Delta s_{t+9} - \Delta l_{t+19} \geq 0, \\ -\Delta s_t - \Delta s_{t+5} - \Delta s_{t+9} + \Delta l_{t+19} + c + 2 \geq 0, \\ \Delta s_t + \Delta s_{t+5} - \Delta s_{t+9} + \Delta l_{t+19} + c \geq 0, \\ \Delta s_t - \Delta s_{t+5} + \Delta s_{t+9} + \Delta l_{t+19} + c \geq 0, \\ -\Delta s_t + \Delta s_{t+5} + \Delta s_{t+9} + \Delta l_{t+19} + c \geq 0, \end{cases}$$

$$L_3 = \begin{cases} -\Delta l_{t+19} - c + 1 \geq 0, \\ -\Delta s_{t+4} + c \geq 0, \\ -\Delta s_{t+7} + c \geq 0, \\ \Delta s_{t+4} + \Delta s_{t+7} - c \geq 0, \\ \Delta s_t - \Delta s_{t+5} + \Delta l_{t+19} + c \geq 0, \\ -\Delta s_t + \Delta s_{t+5} + \Delta l_{t+19} + c \geq 0, \\ \Delta s_t + \Delta s_{t+5} - \Delta l_{t+19} \geq 0, \\ -\Delta s_t - \Delta s_{t+5} - \Delta l_{t+19} + 2 \geq 0. \end{cases} \quad (5)$$

(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 MILP. 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 Section 3.2, we propose a method to evaluate the update bit difference probability, which helps us find the bit whose difference probability deviates significantly from 1/2 and has the largest number of rounds.

3.2. Detecting the Bias of the Difference. In [9, 11], a bias was detected by experimentally observing certain nonrandomness, 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. When conditions cease being applied, we get the probability of difference in each bit of two NLFSRs at that round. Using these probabilities, we can calculate the update bit difference probability in each subsequent round.

Property 1. Let a, b be two independent random Boolean variables, and then, the probability $P\{\Delta(a \oplus b) = 1\} = P\{\Delta a = 1\} + P\{\Delta b = 1\} - 2P\{\Delta a = 1\}P\{\Delta b = 1\}$.

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, and then, the probability

$$\begin{aligned} &P\{\Delta(x \oplus y \oplus z \oplus w) = 1\} \\ &\quad = \frac{1}{2} - \frac{1}{2}(1 - 2P\{\Delta x = 1\})(1 - 2P\{\Delta y = 1\}) \\ &\qquad \times (1 - 2P\{\Delta z = 1\})(1 - 2P\{\Delta w = 1\}). \end{aligned} \quad (6)$$

In the following, we consider the difference probability of two Boolean variables' products. Property 3 shows us how to evaluate the probability.

Property 3. Let a, b be the same as defined in Property 1, and then, the probability

$$\begin{aligned} P\{\Delta(a \cdot b) = 1\} = (&P\{\Delta a = 1\} + P\{\Delta b = 1\} \\ &- P\{\Delta a = 1\}P\{\Delta b = 1\}) \cdot \frac{1}{2}. \end{aligned} \quad (7)$$

In Equations (2) and (3), there is no difference in the key and const, so 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 Equation (2), we can obtain the formula to calculate the probability of $\Delta s_{t+13} = 1$:

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

where

$$\begin{aligned} P\{\Delta(l_{t+6}l_{t+8}) = 1\} = (&P\{\Delta l_{t+6} = 1\} + P\{\Delta l_{t+8} = 1\} \\ &- P\{\Delta l_{t+6} = 1\}P\{\Delta l_{t+8} = 1\}) \cdot \frac{1}{2}, \\ P\{\Delta(l_{t+10}l_{t+15}) = 1\} = (&P\{\Delta l_{t+10} = 1\} + P\{\Delta l_{t+15} = 1\} \\ &- P\{\Delta l_{t+10} = 1\}P\{\Delta l_{t+15} = 1\}) \cdot \frac{1}{2}. \end{aligned} \quad (9)$$

From Equation (3), we can obtain the formula to calculate the probability of $\Delta l_{t+19} = 1$:

$$\begin{aligned} P\{\Delta l_{t+19} = 1\} &= P\{\Delta(s_t \oplus s_{t+5} \oplus s_{t+4}s_{t+7} \oplus s_{t+9}a_t) = 1\} \\ &= \frac{1}{2} - \frac{1}{2} \cdot (1 - 2P\{\Delta s_t = 1\})(1 - 2P\{\Delta s_{t+5} = 1\}) \\ &\quad \cdot (1 - 2P\{\Delta(s_{t+4}s_{t+7}) = 1\}) \\ &\quad \cdot (1 - 2P\{\Delta(s_{t+9}a_t) = 1\}), \end{aligned} \quad (10)$$

Input: $\{P\{\Delta s_t = 1\}, P\{\Delta s_{t+1} = 1\}, \cdots P\{\Delta s_{t+12} = 1\}\}$: the set of probabilities of the difference for each bit of the 13-bit NLFSR at round t; $\{P\{\Delta l_t = 1\}, P\{\Delta l_{t+1} = 1\}, \cdots P\{\Delta l_{t+18} = 1\}\}$: 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.
$S \coloneqq \{P\{\Delta s_t = 1\}, P\{\Delta s_{t+1} = 1\}, \cdots P\{\Delta s_{t+12} = 1\}\}$;
$L \coloneqq \{P\{\Delta l_t = 1\}, P\{\Delta l_{t+2} = 1\}, \cdots P\{\Delta s_{t+18} = 1\}\}$;
$A \coloneqq \varnothing$;
for $i \in \{t, t+1, \cdots, u\}$ **do**
  $P\{\Delta s_{i+13} = 1\} \coloneqq$ the probability calculated from L according to formulas (8) and (9);
  $P\{\Delta l_{i+19} = 1\} \coloneqq$ the probability calculated from S according to formulas (10) and (11);
  $S \coloneqq \{P\{\Delta s_{i+1} = 1\}, P\{\Delta s_{i+2} = 1\}, \cdots P\{\Delta s_{i+13} = 1\}\}$;
  $L \coloneqq \{P\{\Delta l_{i+1} = 1\}, P\{\Delta l_{i+2} = 1\}, \cdots P\{\Delta l_{i+19} = 1\}\}$;
  $A \coloneqq A \cup \{(P\{\Delta s^{-}(i+13) = 1\}, P\{\Delta l^{-}(i+19) = 1\})\}$
end
return A.

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

where

$$P\{\Delta(s_{t+4}s_{t+7}) = 1\} = (P\{\Delta s_{t+4} = 1\} + P\{\Delta s_{t+7} = 1\} - P\{\Delta s_{t+4} = 1\}P\{\Delta s_{t+7} = 1\}) \cdot \frac{1}{2},$$
$$P\{\Delta(s_{t+9}a_t) = 1\} = a_t P\{\Delta s_{t+9} = 1\}. \quad (11)$$

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 the MILP method to KATAN32 for different rounds to obtain different differential characteristics and minimum conditions. We choose 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. However, we cannot 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 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

$$s_{13} = x_{18} \oplus x_7 \oplus x_{12}x_{10} \oplus x_8x_3 \oplus k_1 = 0. \quad (12)$$

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

$$l_{23} = x_{27} \oplus x_{22} \oplus k_8 = 0. \quad (13)$$

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

$$l_{20} = x_{30} \oplus x_{25} \oplus x_{21} \oplus k_2 = 0. \quad (14)$$

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

$$l_{24} = x_{26} \oplus x_{21} \oplus x_{22}x_{19} \oplus x_{17} \oplus x_6 \oplus k_3 \oplus k_{10} = 0. \quad (15)$$

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

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

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

$$l_{31} = x_{19} \oplus x_{14} \oplus x_3 \oplus x_8 x_6 \oplus x_4 (x_{31} \oplus x_{26} \oplus x_{22} \oplus k_0) \oplus (x_{15} \oplus x_4 \oplus k_7)(x_{12} \oplus x_1 \oplus x_6 x_4 \oplus k_{13}) \oplus k_9 \oplus k_{24} = 0. \quad (17)$$

The difference propagation and the conditions applied are presented in Table 6.

After imposing these conditions, we obtain the probability of difference in each bit at round 24 as follows: (0,0,0,0,0,0,0,0, 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).

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. Among the bits whose positions are very close to l_{82}, l_{79} has the maximum biased difference, shown as follows:

$$P\{\Delta l_{79} = 1 \mid \text{all the conditions satisfied}\} \approx 0.5 - 0.00001. \quad (18)$$

We confirmed the strongly biased difference in bit l_{79} experimentally. Let us consider the conditions applied. There are ten 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 the remaining bits are 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 conditions (12)–(17) 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 because not every plaintext can pass the filtering. 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 conditions (12)–(17), 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 ten equivalent key bits. For each guess, similar to the verification, we use conditions (12)–(17) to filter 2^{21} plaintexts of which the 21 bits involved in conditions (12)–(17) 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, and then, $P\{\Delta l_{79} = 1\}$ shows the obvious bias. In the 1024 statistical results from guesses of 10 equivalent key bits, the maximum bias in the results corresponds to the ten equivalent key bits' correct values. 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 choose four 80-bit keys and mount four key-recovery attack experiments and each time the ten equivalent key bits can be recovered correctly, as shown by the results listed in Table 7.

Table 6: Differential characteristic and conditions for $\Delta X = 0xc4200801$.

Round	Difference state	Conditions
0	***1***100***01***0***0***0100 ***0***0000***0***1***0***0***0***0***0***0001	
1	***1***000***1***0***0***1***0***0 ***0***0000***0***1***0***0***0***0***0***0***0***010	$x_9 = 0, x_{23} = 0$
2	***0***001***0***0***0***1***0***000 ***0***0000***0***1***0***0***0***0***0***0***0***0***0***100	
3	**0**010***0***0***0***1***0***0***0***0 ***0***0001***0***0***0***0***0***0***0***0***1***000	$x_5 = 0, x_{24} = 0$
4	***0***100***0***1***0***0***0***0***0 ***0***0010***0***0***0***0***0***0***0***0***1***0***0***0	
5	***1***000***0***1***0***0***0***0***0 ***0***0100***0***0***0***0***0***0***0***0***1***0***0***0	
6	***0***000***1***0***0***0***0***0***0 ***0***1000***0***0***0***0***0***0***0***1***0***0***0***0	Condition (12)
7	***0***001***0***0***0***0***0***0***0 ***1***0000***0***0***0***0***0***1***0***0***0***0***0	
8	***0***010***0***0***0***0***0***0***0 ***0***0000***0***0***0***0***1***0***0***0***0***0***0	Condition (13)
9	***0***100***0***0***0***0***0***0***0 ***0***0000***0***0***0***0***0***1***0***0***0***0***0	
10	***1***000***0***0***0***0***0***0***0 ***0***0000***0***0***0***0***0***1***0***0***0***0***0	$x_2 = 0$
11	***0***000***0***0***0***0***0***0***0 ***0***0000***0***1***0***0***0***0***0***0***0***0***1	
12	***0***000***0***0***0***0***0***0***0 ***0***0000***0***0***0***0***0***0***0***0***1***0	Condition (14)
13	***0***000***0***0***0***0***0***0***0 ***0***0001***0***0***0***0***0***0***0***1***0***0	
14	***0***000***0***0***0***0***0***0***0 ***0***0010***0***0***0***0***0***0***0***1***0***0	Condition (15)
15	***0***000***0***0***0***0***0***0***0 ***0***0100***0***0***0***0***0***0***1***0***0***0	
16	***0***000***0***0***0***0***0***0***0 ***0***1000***0***0***0***0***0***1***0***0***0***0	
17	***0***000***0***0***0***0***0***0***0 ***0***1000***0***0***0***0***0***0***1***0***0***0	
18	***0***000***0***0***0***0***0***0***0 ***1***0000***0***0***0***1***0***0***0***0***0***0	
19	***0***000***0***0***0***0***0***0***0 ***0***0000***0***0***0***0***1***0***0***0***0***0	
20	***0***000***0***0***0***0***0***0***0∗***0***0000***0***0***0***1***0***0***0***0***0***0	
21	***0***000***0***0***0***0***0***0***0∗**0** ***0***0000***0***0***1***0***0***0***0***0***0	Condition (16)
22	***0***000***0***0***0***0***0***0∗**0**0 ***0***0000***0***0***0***1***0***0***0***0***0***0	
23	***0***000***0***0***0***0***0∗**0**0**0** ***0***0000***0***0***1***0***0***0***0***0***0***0	Condition (17)
24	***0***000***0***0***0***0∗***0***0**0** ***0***0000***0***1***0***0***0***0***0***0***0***0	

The bold bits denote the update bits. The bold italic bits denote the bits that generate the update bits. The differential probability of the bit ∗ is 1/2.

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 $\Delta s_{14} = x_2$ and 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

$$l_{27} = x_{23} \oplus x_{18} \oplus x_7 \oplus x_{12} x_{10} \oplus x_8 x_3 \oplus x_{19} (x_{16} \oplus x_{10} x_8 \oplus k_5) \oplus k_{16} \oplus k_1 = 0. \quad (19)$$

Table 7: 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

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

$$l_{20} = x_{30} \oplus x_{25} \oplus x_{26}x_{23} \oplus x_{21} \oplus k_2 = 0. \quad (20)$$

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

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

The differences in propagation and conditions applied are presented in Table 8. After imposing these conditions, we obtain the probability of difference in each bit at round 17 as follows: $(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

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:

$$P\{\Delta l_{81} = 1 \mid \text{all the conditions satisfied}\} \approx 0.5 + 0.000226. \quad (22)$$

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 (19)–(21). We choose 2^4 keys of which k_1, k_2, k_3, k_5 are free and the other bits are 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 (19)–(21) 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 calculate$P\{\Delta l_{81} = 1\}$. 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 (19)–(21), 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 (19)–(21). 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 (19)–(21). For each of the 2^4 guesses of these four variables, we use conditions (19)–(21) 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 calculate $P\{\Delta l_{81} = 1\}$. 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 choose five 80-bit keys randomly and mount 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 9.

Table 8: Differential characteristic and conditions for $\Delta X = 0x10040080$.

Round	Difference state	Conditions
0	***0***001***000000***000 ***1***0000***000001***0000***0***000	
1	***0***010***000000***0**0** ***0***0000***000010***0000***0***00**0**	$x_2 = 0$
2	***0***100***000000***0**0** ***0***0000***000100***0000***0***00**0**	
3	***1***000***000000***0**0** ***0***0000***001000***0000***0***00**0**	$x_9 = 0$
4	***0***000***000000***0**0** ***0***0000***010000***0000***0***001	
5	***0***000***000000***0**0** ***0***0000***100000***0000***0***01**0**	$x_5 = 0$
6	***0***000***000000***0**0** ***0***0001***000000***0000***1***00**0**	
7	***0***000***000000***0**0** ***0***0001***000000***0001***0***00**0**	$x_1 = 0$
8	***0***000***000000***0**0** ***0***0010***000000***0010***0***00**0**	
9	***0***000***000000***0**0** ***0***0100***000000***0100***0***00**0**	
10	***0***000***000000***0**0** ***0***1000***000000***1000***0***00**0**	
11	***0***000***000000***0**0** ***1***0000***000010***0000***0***00**0**	
12	***0***000***000000***0**0** ***0***0000***000100***0000***0***00**0**	Condition (19)
13	***0***000***000000***0**0** ***0***0000***001000***0000***0***00**0**	
14	***0***000***000000***0**0** ***0***0000***010000***0000***0***00**0**	Condition (20)
15	***0***000***000000***0**0** ***0***0000***100000***0000***0***00**0**	
16	***0***000***000000***0**0** ***0***0000***100000***0000***0***00**0**	Condition (21)
17	***0***000***000000***0**0** ***0***0000***100000***0000***0***00**0**	

The bold bits denote the update bits. The bold italic bits denote the bits that generate the update bits.

5. Extension with the Standard Differential Attack

Combined with the standard differential attack, the conditional differential attack on 81-round KATAN32 can be

Table 9: 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

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 [34], we give the algebraic representation of the intermediate state using the ciphertext and round keys.

Using Equations (2) and (3), we can get the expression of l_t, s_t in decryption direction:

$$l_t = s_{t+13} \oplus l_{t+11} \oplus l_{t+6} l_{t+8} \oplus l_{t+10} l_{t+15} \oplus k_{2t+1}, \tag{23}$$

$$s_t = l_{t+19} \oplus s_{t+5} \oplus s_{t+4} s_{t+7} \oplus s_{t+9} a_t \oplus k_{2t}. \tag{24}$$

Suppose the output bits of 97-round KATAN32 corresponding to plaintext X are $S_{97} = (s_{97}, s_{98}, \cdots, s_{110})$ and $L_{97} = (l_{97}, l_{98}, \cdots, l_{115})$, and the output bits of 97-round KATAN32 corresponding to plaintext $X + \Delta X$ are $S'_{97} = (s'_{97}, s'_{98}, \cdots, s'_{110})$ and $L'_{97} = (l'_{97}, l'_{98}, \cdots, l'_{115})$. For decryption direction, Δl_{81} can be expressed by round keys and the ciphertext of 97-round KATAN32 by using Equations (23) and (24) iteratively.

$$\begin{aligned}
\Delta l_{81} = {} & l_{113} \oplus s_{99} \oplus s_{98} s_{101} \oplus s_{105} \oplus l_{103} \oplus l_{98} l_{100} \oplus l_{102} l_{107} \\
& \oplus (s_{100} \oplus l_{98} \oplus (s_{106} \oplus l_{104} \oplus l_{99} l_{101} \oplus l_{103} l_{108} \oplus k_{187}) \\
& \cdot (s_{108} \oplus l_{106} \oplus l_{101} l_{103} \oplus l_{105} l_{110} \oplus k_{191}) \oplus l_{97} l_{102} \oplus k_{175}) \\
& \cdot (s_{102} \oplus l_{100} \oplus (s_{108} \oplus l_{106} \oplus l_{101} l_{103} \oplus l_{105} l_{110} \oplus k_{191}) l_{97} \\
& \oplus l_{99} l_{104} \oplus k_{179}) \oplus (s_{104} \oplus l_{102} \oplus l_{97} l_{99} \oplus l_{101} l_{106} \oplus k_{183}) \\
& \cdot (s_{109} \oplus l_{107} \oplus l_{102} l_{104} \oplus l_{106} l_{111} \oplus k_{193}) \oplus 1'_{113} \oplus s'_{99} \\
& \oplus s'_{98} s'_{101} \oplus s'_{105} \oplus l'_{103} \oplus l'_{98} l'_{100} \oplus l'_{102} l'_{107} \\
& \oplus \left(s'_{100} \oplus l'_{98} \oplus \left(s'_{106} \oplus l'_{104} \oplus l'_{99} l'_{101} \oplus l'_{103} l'_{108} \oplus k_{187}\right)\right. \\
& \cdot \left.\left(s'_{108} \oplus l'_{106} \oplus l'_{101} l'_{103} \oplus l'_{105} l'_{110} \oplus k_{191}\right) \oplus l'_{97} l'_{102} \oplus k_{175}\right) \\
& \cdot \left(s'_{102} \oplus l'_{100} \oplus \left(s'_{108} \oplus l'_{106} \oplus l'_{101} l'_{103} \oplus l'_{105} l'_{110} \oplus k_{191}\right) l'_{97}\right. \\
& \oplus \left. l'_{99} l'_{104} \oplus k_{179}\right) \oplus \left(s'_{104} \oplus l'_{102} \oplus l'_{97} l'_{99} \oplus l'_{101} l'_{106} \oplus k_{183}\right) \\
& \cdot \left(s'_{109} \oplus l'_{107} \oplus l'_{102} l'_{104} \oplus l'_{106} l'_{111} \oplus k_{193}\right).
\end{aligned} \tag{25}$$

According to this expression, one can calculate Δl_{81} by using the ciphertexts of 97-round KATAN32 and six equivalent key bits $k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}$. We extend the attack described in Section 4.2 to 97-round. Plaintexts being filtered by the conditions are encrypted to ciphertexts by 97-round KATAN32. Δl_{81} can be computed from ciphertexts of 97-round KATAN32 and the guess of these six equivalent key bits $k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}$. Given every guess of ten 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, $P\{\Delta l_{81} = 1\}$ shows an obvious bias. The computational cost of the experiment is less than 2^{24+6} encryptions of 97-round KATAN32. We mount five key-recovery attack experiments with the same key as the experiments in Section 4.2, and each time the ten equivalent key bits can be correctly recovered.

5.2. Key-Recovery Attack on 98-Round KATAN32. Suppose the output bits of 98-round KATAN32 corresponding to plaintext X are $S_{98} = (s_{98}, s_{99}, \cdots, s_{111})$ and $L_{98} = (l_{98}, l_{99}, \cdots, l_{116})$, and the output bits corresponding to plaintext $X + \Delta X$ are $S'_{98} = (s'_{98}, s'_{99}, \cdots, s'_{111})$ and $L'_{98} = (l'_{98}, l'_{99}, \cdots, l'_{116})$. For decryption direction, Δl_{81} can be expressed using round keys and the ciphertext of 98-round KATAN32.

$$\begin{aligned}
\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}) \\
& \cdot (s_{108} \oplus l_{106} \oplus l_{101} l_{103} \oplus l_{105} l_{110} \oplus k_{191}) \\
& \oplus (s_{110} \oplus l_{108} \oplus l_{103} l_{105} \oplus l_{107} l_{112} \oplus k_{195}) l_{102} \oplus k_{175}) \\
& \cdot (s_{102} \oplus l_{100} \oplus (s_{108} \oplus l_{106} \oplus l_{101} l_{103} \oplus l_{105} l_{110} \oplus k_{191}) \\
& \cdot (s_{110} \oplus l_{108} \oplus l_{103} l_{105} \oplus l_{107} l_{112} \oplus k_{195}) \oplus l_{99} l_{104} \oplus k_{179}) \\
& \oplus (s_{104} \oplus l_{102} \oplus (s_{110} \oplus l_{108} \oplus l_{103} l_{105} \oplus l_{107} l_{112} \oplus k_{195}) 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 \left(s'_{100} \oplus l'_{98} \oplus \left(s'_{106} \oplus l'_{104} \oplus l'_{99} l'_{101} \oplus l'_{103} l'_{108} \oplus k_{187}\right)\right. \\
& \cdot \left.\left(s'_{108} \oplus l'_{106} \oplus l'_{101} l'_{103} \oplus l'_{105} l'_{110} \oplus k_{191}\right)\right. \\
& \oplus \left.\left(s'_{110} \oplus l'_{108} \oplus l'_{103} l'_{105} \oplus l'_{107} l'_{112} \oplus k_{195}\right) l'_{102} \oplus k_{175}\right) \\
& \cdot \left(s'_{102} \oplus l'_{100} \oplus \left(s'_{108} \oplus l'_{106} \oplus l'_{101} l'_{103} \oplus l'_{105} l'_{110} \oplus k_{191}\right)\right. \\
& \cdot \left.\left(s'_{110} \oplus l'_{108} \oplus 1'_{103} l'_{105} \oplus l'_{107} l'_{112} \oplus k_{195}\right) \oplus l'_{99} l'_{104} \oplus k_{179}\right) \\
& \oplus \left(s'_{104} \oplus l'_{102} \oplus \left(s'_{110} \oplus l'_{108} \oplus l'_{103} l'_{105} \oplus l'_{107} l'_{112} \oplus k_{195}\right) l'_{99}\right. \\
& \oplus \left. l'_{101} l'_{106} \oplus k_{183}\right) \left(s'_{109} \oplus l'_{107} \oplus l'_{102} l'_{104} \oplus l'_{106} l'_{111} \oplus k_{193}\right).
\end{aligned} \tag{26}$$

The expression contains seven equivalent key bits k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}, k_{195}, which makes the computational cost of the key-recovery attack be less than 2^{24+7} times 98-round KATAN32 encryption. 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. Suppose the output bits of 99-round KATAN32 corresponding to plaintext X are $S_{99} = (s_{99}, s_{100}, \cdots, s_{112})$ and $L_{99} = (l_{99}, l_{100}, \cdots, l_{117})$, and the output bits corresponding to plaintext $X + \Delta X$ are $S'_{99} = (s'_{99}, s'_{100}, \cdots, s'_{112})$ and $L'_{99} = (l'_{99}, l'_{100}, \cdots, l'_{117})$. For decryption direction, Δl_{81} can be expressed using round keys and the ciphertext of 99-round KATAN32.

$$\begin{aligned}
\Delta l_{81} = {} & l_{113} \oplus s_{99} \oplus (l_{117} \oplus s_{103} \oplus s_{102}s_{105} \oplus s_{107} \oplus k_{196})s_{101} \oplus s_{105} \\
& \oplus l_{103} \oplus (s_{111} \oplus l_{109} \oplus l_{104}l_{106} \oplus l_{108}l_{113} \oplus k_{197})l_{100} \\
& \oplus l_{102}l_{107} \oplus (s_{100} \oplus (s_{111} \oplus l_{109} \oplus l_{104}l_{106} \oplus l_{108}l_{113} \oplus k_{197}) \\
& \oplus (s_{106} \oplus l_{104} \oplus l_{99}l_{101} \oplus l_{103}l_{108} \oplus k_{187}) \\
& \cdot (s_{108} \oplus l_{106} \oplus l_{101}l_{103} \oplus l_{105}l_{110} \oplus k_{191}) \\
& \oplus (s_{110} \oplus l_{108} \oplus l_{103}l_{105} \oplus l_{107}l_{112} \oplus k_{195})l_{102} \oplus k_{175}) \\
& \cdot (s_{102} \oplus l_{100} \oplus (s_{108} \oplus l_{106} \oplus l_{101}l_{103} \oplus l_{105}l_{110} \oplus k_{191}) \\
& \cdot (s_{110} \oplus l_{108} \oplus l_{103}l_{105} \oplus l_{107}l_{112} \oplus k_{195}) \oplus l_{99}l_{104} \oplus k_{179}) \\
& \oplus (s_{104} \oplus l_{102} \oplus (s_{110} \oplus l_{108} \oplus l_{103}l_{105} \oplus l_{107}l_{112} \oplus k_{195})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 \left(l'_{117} \oplus s'_{103} \oplus s'_{102}s'_{105} \oplus s'_{107} \oplus k_{196}\right)s'_{101} \\
& \oplus s'_{105} \oplus l'_{103} \oplus \left(s'_{111} \oplus l'_{109} \oplus l'_{104}l'_{106} \oplus l'_{108}l'_{113} \oplus k_{197}\right)l'_{100} \\
& \oplus l'_{102}l'_{107} \oplus \left(s'_{100} \oplus \left(s'_{111} \oplus l'_{109} \oplus l'_{104}l'_{106} \oplus l'_{108}l'_{113} \oplus k_{197}\right)\right. \\
& \oplus \left(s'_{106} \oplus l'_{104} \oplus l'_{99}l'_{101} \oplus l'_{103}l'_{108} \oplus k_{187}\right) \\
& \cdot \left(s'_{108} \oplus l'_{106} \oplus l'_{101}l'_{103} \oplus l'_{105}l'_{110} \oplus k_{191}\right) \\
& \oplus \left(s'_{110} \oplus l'_{108} \oplus l'_{103}l'_{105} \oplus l'_{107}l'_{112} \oplus k_{195}\right)l'_{102} \oplus k_{175}\Big) \\
& \cdot \left(s'_{102} \oplus l'_{100} \oplus \left(s'_{108} \oplus l'_{106} \oplus l'_{101}l'_{103} \oplus l'_{105}l'_{110} \oplus k_{191}\right)\right. \\
& \cdot \left(s'_{110} \oplus l'_{108} \oplus l'_{103}l'_{105} \oplus l'_{107}l'_{112} \oplus k_{195}\right) \oplus l'_{99}l'_{104} \oplus k_{179}\Big) \\
& \oplus \left(s'_{104} \oplus l'_{102} \oplus \left(s'_{110} \oplus l'_{108} \oplus l'_{103}l'_{105} \oplus l'_{107}l'_{112} \oplus k_{195}\right)l'_{99}\right. \\
& \oplus l'_{101}l'_{106} \oplus k_{183}\Big)\left(s'_{109} \oplus l'_{107} \oplus l'_{102}l'_{104} \oplus l'_{106}l'_{111} \oplus k_{193}\right).
\end{aligned} \tag{27}$$

There are nine equivalent key bits k_{175}, k_{179}, k_{183}, k_{187}, k_{191}, k_{193}, k_{195}, k_{196}, k_{197}. So the computational cost of the key-recovery attack is less than 2^{24+9} times 99-round KATAN32 encryption. 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 the computational cost of less than 2^{63} times 114-round KATAN32 encryption.

6. Conclusion

Conditional differential analysis towards the NLFSR is quite a recent research topic. We advance the research in this direction by using Mixed Integer Linear Programming on the NLFSR-based block cipher KATAN32, a newly typical and well-designed lightweight block cipher. It is the first time applying MILP in the automatically searching for conditional differential trails. Using MILP helps us efficiently obtain the initial difference and conditions of the conditional differential analysis. We propose a new method to quickly calculate the probability of the difference to detect the bit with a bias. We apply the improved conditional differential analysis to KATAN32 and obtain two results, recovering ten equivalent key bits of 79-round KATAN32 and four equivalent key bits of 81-round KATAN32, respectively.

Combined with the 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 the previously best practical distinguisher on KATAN32, our results are extended more than seven rounds with less 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.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61672330 and 11771256).

References

[1] D. Wang and P. Wang, "Two birds with one stone: two-factor authentication with security beyond conventional bound," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 708–722, 2018.

[2] D. Wang, W. Li, and P. Wang, "Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4081–4092, 2018.

[3] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K. R. Choo, "Unified biometric privacy preserving three-factor authentication and key agreement for cloud-assisted autonomous vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9390–9401, 2020.

[4] C. Wang, D. Wang, Y. Tu, G. Xu, and H. Wang, "Understanding node capture attacks in user authentication schemes for

wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2020.

[5] C. D. Cannière, O. Dunkelman, and M. Knezevic, "KATAN and KTANTAN – a family of small and efficient hardware-oriented block ciphers," in *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings, ser. Lecture Notes in Computer Science*, C. Clavier and K. Gaj, Eds., vol. 5747, pp. 272–288, Springer, 2009.

[6] G. Han and W. Zhang, "Improved biclique cryptanalysis of the lightweight block cipher piccolo," *Security and Communication Networks*, vol. 2017, 12 pages, 2017.

[7] M. Liu, "Degree evaluation of nfsr-based cryptosystems," in *Advances in Cryptology - CRYPTO 2017-37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III, ser. Lecture Notes in Computer Science*, J. Katz and H. Shacham, Eds., vol. 10403, pp. 227–249, Springer, 2017.

[8] L. Wei, C. Rechberger, J. Guo, H. Wu, H. Wang, and S. Ling, "Improved meet- in-the-middle cryptanalysis of KTANTAN (poster)," in *Information Security and Privacy -16th Australasian Conference, ACISP 2011, Melbourne, Australia, July 11-13, 2011. Proceedings, ser. Lecture Notes in Computer Science*, U. Parampalli and P. Hawkes, Eds., vol. 6812, pp. 433–438, Springer, 2011.

[9] S. Knellwolf, W. Meier, and M. Naya-Plasencia, "Conditional differential cryptanalysis of nlfsr-based cryptosystems," in *Advances in Cryptology - ASIACRYPT 2010 -16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings, ser. Lecture Notes in Computer Science*, M. Abe, Ed., vol. 6477, pp. 130–145, Springer, 2010.

[10] I. Ben-Aroya and E. Biham, "Differtial cryptanalysis of lucifer," in *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings, ser. Lecture Notes in Computer Science*, D. R. Stinson, Ed., vol. 773, pp. 187–199, Springer, 1993.

[11] S. Knellwolf, W. Meier, and M. Naya-Plasencia, "Conditional differential cryptanalysis of trivium and KATAN," in *Selected Areas in Cryptography -18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers, ser. Lecture Notes in Computer Science*, A. Miri and S. Vaudenay, Eds., vol. 7118, pp. 200–212, Springer, 2011.

[12] M. R. Albrecht and G. Leander, "An all-in-one approach to differential cryptanalysis for small block ciphers," *IACR Cryptology ePrint Archive*, vol. 2012, article 401, 2012.

[13] T. Isobe and K. Shibutani, "Improved all-subkeys recovery attacks on fox, KATAN and SHACAL-2 block ciphers," in *Fast Software Encryption -21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers, ser. Lecture Notes in Computer Science*, C. Cid and C. Rechberger, Eds., vol. 8540, pp. 104–126, Springer, 2014.

[14] T. Fuhr and B. Minaud, "Match box meet-in-the-middle attack against KATAN," in *Fast Software Encryption -21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers, ser. Lecture Notes in Computer Science*, C. Cid and C. Rechberger, Eds., vol. 8540, pp. 61–81, Springer, 2014.

[15] Z. Ahmadian, S. Rasoolzadeh, M. Salmasizadeh, and M. R. Aref, "Automated dynamic cube attack on block ciphers: cryptanalysis of SIMON and KATAN," *IACR Cryptology ePrint Archive*, vol. 2015, 2015.

[16] B. Zhu and G. Gong, "Multidimensional meet-in-the-middle attack and its applications to KATAN32/48/64," *Cryptography and Communications*, vol. 6, no. 4, pp. 313–333, 2014.

[17] S. Rasoolzadeh and H. Raddum, "Multidimensional meet in the middle cryptanalysis of KATAN," *IACR Cryptology ePrint Archive*, vol. 2016, 2016.

[18] T. Isobe, "A single-key attack on the full GOST block cipher," *Journal of Cryptology*, vol. 26, no. 1, pp. 172–189, 2013.

[19] Z. Jiang and C. Jin, "Impossible differential cryptanalysis of 8-round deoxys bc-256," *IEEE Access*, vol. 6, pp. 8890–8895, 2018.

[20] W. Zhang, Y. Li, and L. Wu, "A new one-bit difference collision attack on HAVAL-128," *Science China Information Sciences*, vol. 55, no. 11, pp. 2521–2529, 2012.

[21] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "Differential and linear cryptanalysis using mixed-integer linear programming," in *Information Security and Cryptology -7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers, ser. Lecture Notes in Computer Science*, C. Wu, M. Yung, and D. Lin, Eds., vol. 7537, pp. 57–76, Springer, 2011.

[22] S. Wu and M. Wang, "Security evaluation against differential cryptanalysis for block cipher structures," *IACR Cryptology ePrint Archive*, vol. 2011, article 551, 2011.

[23] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic security evaluation and (related-key) differential characteristic search: application to simon, present, lblock, DES(L) and other bit-oriented block ciphers," in *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I, ser. Lecture Notes in Computer Science*, P. Sarkar and T. Iwata, Eds., vol. 8873, pp. 158–178, Springer, 2014.

[24] P. Zhang and W. Zhang, "Differential cryptanalysis on block cipher skinny with MILP program," *Security and Communication Networks*, vol. 2018, 11 pages, 2018.

[25] Z. Xiang, W. Zhang, Z. Bao, and D. Lin, "Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers," in *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I, ser. Lecture Notes in Computer Science*, J. H. Cheon and T. Takagi, Eds., vol. 10031, pp. 648–678, 2016.

[26] W. Zhang and V. Rijmen, "Division cryptanalysis of block ciphers with a binary diffusion layer," *IET Information Security*, vol. 13, no. 2, pp. 87–95, 2019.

[27] Y. Sasaki and Y. Todo, "New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers," in *Advances in Cryptology - EUROCRYPT 2017 -36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III, ser. Lecture Notes in Computer Science*, J. Coron and J. B. Nielsen, Eds., vol. 10212, pp. 185–215, 2017.

[28] G. Han, W. Zhang, and H. Zhao, "An upper bound of the longest impossible differentials of several block ciphers," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 1, pp. 435–451, 2019.

[29] Z. Li, W. Bi, X. Dong, and X. Wang, "Improved conditional cube attacks on keccak keyed modes with MILP method," in *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I, ser. Lecture Notes in Computer Science*, T. Takagi and T. Peyrin, Eds., vol. 10624, pp. 99–127, Springer, 2017.

[30] L. Song, J. Guo, and D. Shi, "New MILP modeling: improved conditional cube attacks to keccak-based constructions," *IACR Cryptology ePrint Archive*, vol. 2017, 2017.

[31] S. Huang, X. Wang, G. Xu, M. Wang, and J. Zhao, "Conditional cube attack on reduced-round keccak sponge function," in *Advances in Cryptology- EUROCRYPT 2017 -36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II, ser. Lecture Notes in Computer Science*, J. Coron and J. B. Nielsen, Eds., vol. 10211, pp. 259–288, 2017.

[32] E. Biham and O. Dunkelman, "Differential cryptanalysis in stream ciphers," *IACR Cryptology ePrint Archive*, vol. 2007, 2007.

[33] C. D. Cannière, "Analysis of grain's initialization algorithm," in *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings, ser. Lecture Notes in Computer Science, S. Vaudenay*, vol. 5023, pp. 276–289, Springer, 2008.

[34] W. Zhang, M. Cao, J. Guo, and E. Pasalic, "Improved security evaluation of SPN block ciphers and its applications in the single-key attack on SKINNY," *IACR Transactions on Symmetric Cryptology*, vol. 2019, no. 4, pp. 171–191, 2019.