DEEPAND: In-Depth Modeling of Correlated AND Gates for NLFSR-based Lightweight Block Ciphers

Amit Jana[1], Mostafizar Rahman[1] and Dhiman Saha[2]

[1] Indian Statistical Institute, Kolkata
mrahman454@gmail.com,janaamit001@gmail.com
[2] de.ci.phe.red Lab, Department of Electrical Engineering and Computer Science, Indian Institute of Technology Bhilai
dhiman@iitbhilai.ac.in

Abstract. Automated cryptanalysis has taken center stage in the arena of cryptanalysis since the pioneering work by Mouha *et al.* which showcased the power of Mixed Integer Linear Programming (MILP) in solving crypto problems that otherwise required significant effort. Since this inception, research in this area has moved in primarily two directions. One is to model more and more classical cryptanalysis tools as an optimization problem to leverage the ease provided by state-of-the-art solvers. The other direction is to improve existing models to make them more efficient and/or accurate. The current work is an attempt to contribute to the latter. In this work, a general model referred to as DEEPAND has been devised to capture the correlation between AND gates in NLFSR-based lightweight block ciphers. DEEPAND builds upon and generalizes the idea of joint propagation of differences through AND gates captured using refined MILP modeling of TinyJAMBU by Saha *et al.* in FSE 2020. The proposed model has been applied to TinyJAMBU and KATAN and can detect correlations that were missed by earlier models. This leads to more accurate differential bounds for both the ciphers. In particular, a 384-round Type 4 trail is found for TinyJAMBU with 14-active AND gates using the new model, while the refined model reported this figure to be 19. Moreover, we have found a full round Type 4 trail of TinyJAMBU keyed permutation P_{1024} with probabilty $2^{-108} (\gg 2^{-128})$, which violates designer's security claim. Thus, our results shows that TinyJAMBU's underlying keyed-permutation have non-random properties. As a result, it cannot be expected to provide the same security levels as robust block ciphers and also, the provable security of TinyJAMBU AEAD scheme should be carefully revisited. Similarly, for KATAN32, DEEPAND modeling improves the 42-round trail with 2^{-11} probability to 2^{-7}. DEEPAND seems to capture the underlying correlation better when multiple AND gates are at play and can be adapted to other classes of ciphers as well.

Keywords: MILP · KATAN · TinyJAMBU · Symmetric-Key Cryptanalysis

1 Introduction

One of the fundamental decisions in any iterative block cipher design, once we have a *good* round function, is the number of rounds. This decision is a trade-off between security and efficiency and plays an even more critical part in the context of Lightweight Cryptography which is referred to as crypto tailored for resource contained environments. A typical way to decide this is to take into account the penetration of best attack available and then adding some more rounds as the so-called *security-margin*. Traditionally, designers try to prove how many rounds are sufficient to resist a certain kind of attack. This in general is a rigorous task and primarily limited to a specific construction. For instance resistance against differential cryptanalysis [3] relies on the number of active sboxes in the best available differential trail. It has been a long standing question if these seeming critical task of cryptanalysis could be automated or aided in some generic way. Though there have been initial attempts in this direction but the first major breakthrough in this direction is attributed to Mouha *et al.* [7] who was one of the first to demonstrate how the cryptanalytic problem of determining minimum number of active sboxes could be modeled as an optimization problem which could in turn be solved by automated solvers. In particular, the authors showcased how Mixed Integer Linear Programming (MILP) can be leveraged as an ingenious cryptanalysis aid. This seminal work spawned an entirely new line of research where the goal is at one hand to increase the breadth of the strategy with new modelings (applications to linear, division, impossible differential cryptanalysis). On the other had the idea to improve upon the models to capture the underlying crypto property as closely as possible. The current aims to add to state-of-art is better MILP modeling.

Interestingly, researchers have shown that there are mechanism to precisely model valid transition for any crypto property [11, 10, 9]. However, the catch is that these leads to over constrained model that is infeasible to solved in reasonable time. On the other end of the spectrum is an over simplified model which might lead to invalid transitions. There is a rich body of work that tries to reach a middle ground [4, 2]. In FSE 2020, Saha *et al.* made an interesting observation in this line of balanced modeling for the NIST-LWC [] competition finalist TinyJAMBU [1]. The authors pointed out that correlation between multiple AND gates could lead to them becoming dependent leading to joint propagation of differential characteristics. This implies that simple AND gate modeling which treats evey AND gate as an independent entity would produce loose lower bound of minimum number of AND gates. This work shows that this refinement showed by Saha *et al.* can be refined further and a generalized model can be devised to handle the class of Non-Linear Feedback Shift Register (NLFSR) based lightweight block ciphers.

In past decades, several numbers of NLFSR-based block/stream ciphers like Grain , Trivium , KATAN have made great attention in our community in terms of security as well as for the resource constraint environments. Also, this kind of NLFSR-based keyed-permutation is directly used to design lightweight sponge-like authenticated encryption (AE) schemes. TinyJAMBU, designed by Wu and Huang [1] is one of such NLFSR-based sponge-like AE schemes, which is currently one of the ten finalists in the ongoing NIST lightweight competition. The NLFSR-based permutation of TinyJAMBU uses only one NAND gate and some tapin bits to produce a feedback bit which fed into the most significant bit (MSB) of the state, and then apply a one bit shift operation. Recently, in [8], Saha *et al.* have developed a new MILP model by taking an account of the first-order correlation of AND gates, where two subsequent AND computations with a common input position, i.e., the middle bit position b out of three inputs a, b, c to the subsequent ANDs. In this work, they have shown that a correlation betweeen multiple AND gates have a significant impact on the actual probabilities of the differential trails. More secifically, the common input position in the two subsequent ANDs will be revealed when a particlar difference pattern (i.e., $(\Delta a, \Delta b, \Delta c) = (1, 0, 1)$) occur, i.e., for this $(1, 0, 1)$ case, one have to pay a probability for only the first AND whereas the second AND will pass freely. We further reinvestigate this case and observe that due to the difference $(\Delta a, \Delta b) = (1, 0)$, the outut difference (Δz_1) of the first AND directly reveals the bit b, i.e., $\Delta z_1 = b$. Once Δz_1 is fixed, the second AND will be passed freely. In another way, for an AND gate with two inuts a, b, if we know the bit value of a, then for a given difference pattern $(\Delta a, \Delta b) = (0, 1)$, the output difference $\Delta z = a$ will become deterministic. We also observed that, for any NLFSR-based cipher, some AND gates can be freely passed by fixing some particular message bits in the initial state as we know both the input values of the AND computation.

However, these observations are only restricted to the single AND NLFSR-based cipher. For any NLFSR where multiple ANDs are being used inside it, we extend these observations to make more cases (Like (1,0,1)) to increase the probability of a differential trail. Taking all these into account, we have developed a new refined MILP model to generate more optimal differential trails. Finally, we apply this new model in the keyed-permutation of TinyJAMBU AE and to all the KATAN-variants and show that our model captures all possible correlations between ANDs and provides a better optimal differential trails in compared to previous models.

1.1 Our Contributions

This paper attempts to redescribe the two subsequent correlated AND gates $\mathbf{A_1}(a,b)$, $\mathbf{A_2}(b,c)$ as first observed by Saha *et al.* in a general settings. Our first observation is that for a AND gate, if the first input bit of (a,b) (i.e., a) is fixed and the corresponding input bit difference is $(\Delta a, \Delta b) = (0,1)$ then its output difference Δz behaves non-uniformly. Similarly, if b is known and $(\Delta a, \Delta b) = (1,0)$, it follows a non-uniform distribution. Based on these two observation, we have developed a generalized MILP model for differential cryptanalysis, referred to as DEEPAND which captures all possible correlations between multiple ANDs in NLFSR based lightweight block ciphers. Further, we have applied this model to the keyed permutation of TinyJAMBU and all the variants of KATAN , and significantly we have found a better differential bounds. Our contributions can be summarized as follows:

1. Introducing a new generalized MILP model for a class of NLFSR-based ciphers.
2. Applying the model to the KATAN family of ciphers with improved differential bounds for most of the previous attacks reported in literature.
3. Applying the same model to achieve the best bound till date for the NIST LWC finalist TinyJAMBU.
4. Presenting the new model in a generic way to possibly increase its scope for refining MILP model to other class of NLFSR-based stream cipher as well.

1.2 Outline of the Paper

This paper is organized as follows. First, the basic notations, and then the description of TinyJAMBU and KATAN are given in Section 2. In Section 3, we revisit the correlation between two subsequent AND gates in the previous refined MILP model and further, we have shown some observations regarding the non-uniform behaviour of the output distribution of the AND gate. Based on our observations, a new MILP model for a class of NLFSRs with a single/multiple AND in the feedback function to efficiently search for differential trails is explained in Section 4. Our results on differential cryptanalysis for the keyed permution of TinyJAMBU and KATAN family of ciphers is described in Section 5 and Section 6, respectively. Finally, the concluding remarks are furnished in Section 7.

2 Preliminaries

In this section, first of all, the notations used in the paper is described. Then a brief description about TinyJAMBU and KATAN have been provided.

2.1 Notations

The following notations are used throughout the paper.

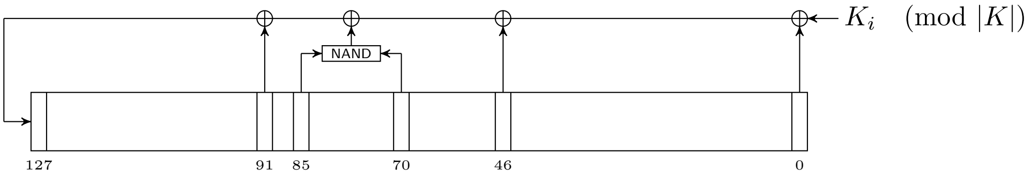

Figure 1: The Permutation P^{k_i}

- $\mathbb{F}_2 = \{0, 1\}$ denotes the finite field with two elements.
- Any element $v \in \mathbb{F}_2^n$ can be represented as a vector or bit string of length n, i.e., $\mid v \mid = n$.
- For $a, b \in \mathbb{F}_2$, $a \oplus b, ab, \overline{a}$ denote the exclusive-or, logical AND, and logical negation, respectively.
- A NAND gate takes input bits a, b, and output as $ab \oplus 1$.
- A n-bit state s can be represented as $(s_{n-1}, s_{n-2}, \cdots, s_1, s_0)$.

2.2 TinyJAMBU

TinyJAMBU is a small variant of JAMBU, is a family of authenticated encryption with associated data (AEAD) schemes. Recently, it is selected as one of the ten finalists among 56 submissions in the NIST Lightweight Cryptography (LWC) Standardization process. In this paper, we are interested to analyze the differential property of the underlying keyed permutation used in TinyJAMBU. The 128-bit keyed permutation P_l^K consists of l number of rounds, where $K \in \mathbb{F}_2^{|K|}$ represents the secret key $(k_{|K|-1}, k_{|K|-2}, \cdots, k_1, k_0)$. Also, for any key K, we use $\mathcal{P}_\mathrm{l}$ to denote an l-round keyed permutation of TinyJAMBU throughout the paper. The $i - th$ round function of the permutation $P^{k_i} : \mathbb{F}_2^{128} \to \mathbb{F}_2^{128}$ transforms a state $(s_{127}, s_{126}, \cdots, s_1, s_0)$ to $(s_f, s_{127}, s_{126}, \cdots, s_2, s_1)$ with $s_f = s_0 \oplus s_{47} \oplus \overline{s_{70}s_{85}} \oplus s_{91} \oplus k_{i \mod |K|}$. The sketch of this permutation is depicted in Figure 1. The l-round transformation of TinyJAMBU with a given key is computed as follows:

$$\prod_{i=0}^{l-1} P^{k_i} = P^{k_{l-1}} \circ P^{k_{l-2}} \circ \cdots \circ P^{k_1} \circ P^{k_0}.$$

For a given secret key K, TinyJAMBU is the sponge-like design that takes a message M, a nonce N, and any associated data A as inputs and finally, outputs a ciphertext C and an authentication tag T. The encryption algorithm of TinyJAMBU can be divided into four phases: initialization, where the state is initialized by sequentially performing the permutation $P_{l_1}^K$ using the key K and nonce N, associated data processing, where each of these blocks of data is sequentially absorbed into the state by using the permutation, encryption phase, where each block of message are sequentially absorbed into the state and parallelly, squeeze another block to output a ciphertext by using the permutation calling, finalization, where it sequentially squeeze a block of data using different rounds permutation call to collect full-length tag. With three different key sizes $K = 128, 192, 256$, TinyJAMBU have three variants. Further, it uses different round numbers to permute its state. Specifically, it uses less number of rounds $(P_{l_1}^K)$ for initialization and associated data processing as compared to message processing $(P_{l_2}^K)$ in the encryption phase, where $l_1 < l_2$. More details can be found in [1].

Table 1: Parameters of KATAN Variants

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

2.3 KATAN

The KATAN family is a very efficient NLFSR-based hardware-oriented block cipher with three variants, namely KATAN32 , KATAN48 , KATAN64 correspond to 32, 48, and 64-bit block sizes. All these variants have 254 rounds and use the non-linear functions NF_1 and NF_2. Also, they use the same LFSR-based key schedule which takes an 80-bit key as an input. The general structure of the KATAN cipher is as follows. First, the plaintext is loaded into two registers L_1 and L_2. In each round, several bits are taken from the registers to fed into the non-linear functions, and finally, the output of NF_1 and NF_2 is loaded to the least significant bits to the registers. The key schedule function expands an 80-bit user-provided key k_i $(0 \leq i < 80)$ into a 508-bit subkey sk_i $(0 \leq i < 508)$ by the following linear operations,

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

Also, the two non-linear functions are defined as follows:

$$NF_1(L_1) = L_1[x_1] \oplus L_1[x_2] \oplus (L_1[x_3] \cdot L_1[x_4]) \oplus (L_1[x_5] \cdot IR) \oplus k_a$$

$$NF_2(L_2) = L_2[y_1] \oplus L_2[y_2] \oplus (L_2[y_3] \cdot L_2[y_4]) \oplus (L_2[y_5] \cdot L_2[y_6])) \oplus k_b,$$

where IR is the round constant value defined in the specifica-IR is the round constant value defined in the specification, and k_a, k_b are the two subkey bits. The selection of the bits $x_i, 1 \leq i \leq 5$ and $y_i, 1 \leq i \leq 6$ are defined for each variant independently, and are listed in Table 1. For KATAN32 , the i-th round function is depicted in Figure 2, where $k_a \leftarrow k_{2i}$ and $k_b \leftarrow k_{2i+1}$. Finally, after 254 rounds, the values of registers are output as a ciphertext. For KATAN48 , the non-linear functions LF_1 and LF_2 are applied twice in one round of the cipher, i.e., the first pair of LF_1 and LF_2 is applied, and then after the update of the registers, they have applied again using the same subkeys. Similarly, in KATAN64 , each round applies LF_1 and LF_2 three times with the same key bits. More details about the specification of KATAN-family of ciphers can be found in [5].

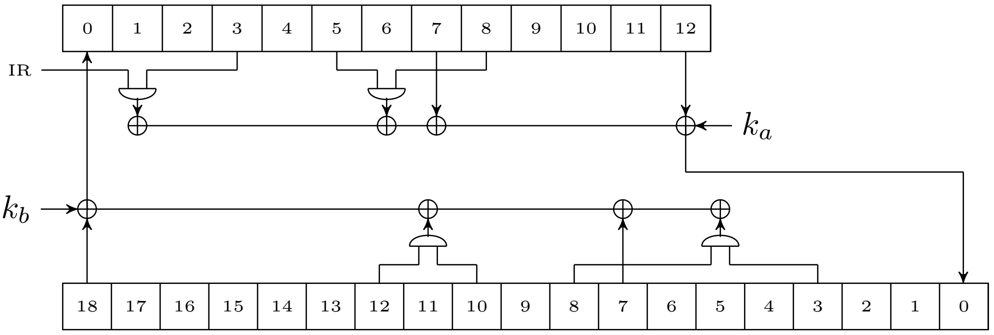

Figure 2: Round Function of KATAN32

Table 2: Difference Distribution of **AND** Gate

a	b	Δa	Δb	Δz
0	0	0	0	0
		0	1	0
		1	0	0
		1	1	1
0	1	0	0	0
		0	1	0
		1	0	1
		1	1	0
1	0	0	0	0
		0	1	1
		1	0	0
		1	1	0
1	1	0	0	0
		0	1	1
		1	0	1
		1	1	1

3 Revisiting the Refined Modeling of AND Gate

Consider an AND gate $\mathbf{A_1}$ with (a, b) as its input, $(\Delta a, \Delta b)$ as its input difference, and Δz as its output difference. Then, the output difference Δz can be expressed as shown in Equation 1.

$$\begin{aligned} \Delta z &= A_1(a, b) \oplus A_1(a \oplus \Delta a, b \oplus \Delta b) \\ &= (a \cdot b) \oplus (a \oplus \Delta a) \cdot (b \oplus \Delta b) \\ &= a \cdot \Delta b \oplus b \cdot \Delta a \oplus \Delta a \cdot \Delta b \end{aligned} \tag{1}$$

Also, the distribution of Δz corresponding to all values of (a, b) and $(\Delta a, \Delta b)$, is shown in Table 2. It is clear from the above table that for a given non-zero input difference $(\Delta a, \Delta b)$ of $\mathbf{A_1}$, $\Pr(\Delta z = 0) = \Pr(\Delta z = 1) = 2^{-1}$, i.e., it behaves uniformly. But, when we give condition on both (a, b) and $(\Delta a, \Delta b)$, then the output difference Δz behaves non-uniformly sometimes. An example for this non-uniform behavior is shown in Example 1. From Table 2, the following observations have been made (Observation 1 may seem trivial but it has been included for the sake of completeness).

Example 1. $\Pr[\Delta z = 0 | (a = 0, \Delta a = 0, \Delta b = 1)] = 1$

Observation 1. *If the value of a, b, Δa, and Δb are known, then Δz becomes deterministic.*

Observation 2. *If $\Delta a = 0$, $\Delta b = 1$ and the value of a is known, then Δz can be determined with probability 1. Similarly, if $\Delta a = 0$, $\Delta b = 1$ and the value of Δz is known, then a can be guessed deterministically.*

Explanation : If $\Delta a = 0$, $\Delta b = 1$, then from Equation 1, $\Delta z = a$. This means, for an input difference $(\Delta a, \Delta b) = (0, 1)$ to $\mathbf{A_1}$, if a is known, then Δz is also known and vice versa.

Observation 3. *If $\Delta a = 1$, $\Delta b = 0$ and the value of b is known then Δz can be determined with probability 1. Similarly, if $\Delta a = 1$, $\Delta b = 0$ and the value of Δz is known, then b can be guessed deterministically.*

Explanation : The explanation is similar to the explanation of Observation 2.

Based on the above observations from Table 2, the distribution Δz of $\mathbf{A_1}$ is directly depends on the input bits a, b when the input difference $(\Delta a, \Delta b)$ is fixed. In another way, according to equation 1, the Δz can be re-written in the following way.

$$\Delta z = \begin{cases} 0, & \text{if } (\Delta a, \Delta b) = (0,0), \\ a, & \text{if } (\Delta a, \Delta b) = (0,1), \\ b, & \text{if } (\Delta a, \Delta b) = (1,0), \\ a \oplus b \oplus 1, & \text{if } (\Delta a, \Delta b) = (1,1), \end{cases}$$

3.1 Related Gates

Consider two subsequent `AND` gates $\mathbf{A_1}$ with (a, b), and $\mathbf{A_2}$ with (b, c) as their inputs, i.e., they both share a common input as b. Also, let $(\Delta a,\ \Delta b)$, $(\Delta b,\ \Delta c)$ are the input differences, and $\Delta z_1, \Delta z_2$ are the output differences of $\mathbf{A_1}, \mathbf{A_2}$ respectively. In [8], Saha *et al.* observed that when $(\Delta a, \Delta b) = (1, 0)$ and $(\Delta b, \Delta c) = (0, 1)$ happens for $\mathbf{A_1}$ and $\mathbf{A_2}$, then $\Delta z_1 = \Delta z_2 = b$. This show that, for two correlated `AND` gates $\mathbf{A_1}$ and $\mathbf{A_2}$, when $(\Delta a, \Delta b, \Delta b) = (1, 0, 1)$ happens, then both the differences are either become 0 with probability 2^{-1} or 1 with probability 2^{-1}. Whereas, for two un-correlated `AND` gates, we have to pay a probability of 2^{-2}. In Lemma 1, this same observation for two correlated `AND` gates can be redescribed on the basis of Observation 2 and Observation 3.

Lemma 1. *Let the input difference to two correlated* `AND` *gates are* $(\Delta a,\ \Delta b)$ *and* $(\Delta b,\ \Delta c)$ *respectively and corresponding output differences are* Δz_1 *and* Δz_2 *respectively. If* $\Delta a = 1$, $\Delta b = 0$, $\Delta c = 1$, *then* $\Pr[\Delta z_1 = \Delta z_2] = 2^{-1}$.

Proof. First of all, the value of Δz_1 is computed first. Thus, for $(\Delta a, \Delta b) = (1, 0)$, it can be concluded that $\Delta z_1 = b$ according to Observation 3. Also, for the second `AND` gate with $(\Delta b, \Delta c) = (0, 1)$, $\Delta z_2 = b$ (from Observation 2). Hence, we have, $\Pr[\Delta z_1 = \Delta z_2] = \Pr(b) = 2^{-1}$. □

Note that, in [8] only Lemma 1 is used; Observation 2 and Observation 3 are not exploited. In this work, these two observations along with Observation 1 are exploited to penetrate more number of rounds for nonlinear-feedback shift register (`NLFSR`) based ciphers.

4 Attack on Nonlinear-Feedback Shift Register (NLFSR) based Ciphers

A nonlinear-feedback shift register (`NLFSR`) is a shift register whose input bit, often called a feedback bit, is a non-linear function of its previous state. In this section, we first review some different class of `NLFSR`s based on the number of `AND` gates it uses to define a non-linear feedback function. We then give the explicit form of theses `NLFSR`s. Finally, we give a general attack framework to capture some correlation among multiple gates.

4.1 NLFSR-based Ciphers with one `AND` Gates in the Feedback Function

Any n-bit cipher based on the `NLFSR`-based keyed permutation with single `AND` gate can be further classified into two cases. In each round of the cipher, the first one is to feed the the feedback bit using non-linear function to the most significant bit (msb) in the state and then shift each bit towards the least significant bit (lsb) (see Figure 1). Similarly, for the second one, compute the feedback bit and feed into the lsb and then shift each bit towards msb. We now give the explicit form of these two NLFSRs.

4.1.1 Computing Forward Differential

Consider an n-bit NLFSR-based cipher $\mathcal{C}$ with s^0 being its initial state value, where $s^0 = (s^0_0, s^0_1, \cdots, s^0_{n-1})$. Then, for each round number $i, 1 \leq i \leq l$, the feedback bit f^i is computed first, in the following way:

$$f^i \leftarrow s_0^{i-1} \oplus s_{j_1}^{i-1} \oplus \cdots \oplus s_{j_m}^{i-1} \oplus s_{u_1}^{i-1} s_{u_2}^{i-1} \oplus K_{(i-1) \mod |K|}.$$

where $0, j_1, \cdots, j_m$ are the tap bit positions of the NLFSR and u_1, u_2 $(u_1 < u_2)$ are the input bits to the AND gate. Then, the state bits in the next round (round $i + 1$) are updated as follows:

$$s_j^i = \begin{cases} s_{j+1}^{i-1}, & \text{for } 0 \leq j \leq (n-2) \\ f^i, & \text{for } j = n-1 \end{cases}$$

Consider a similar cipher $\mathcal{C}'$, whose tap bits are the same as that of $\mathcal{C}$. The only difference is that the bits are shifted in opposite direction as that of $\mathcal{C}$ and in the feedback function s_{n-1}^{i-1} is XOR-ed instead of s_0^{i-1}. The cipher $\mathcal{C}'$ is called *reverse-fed* cipher of $\mathcal{C}$. The feedback bit f^i for $\mathcal{C}'$ is computed as follows:

$$f^i = s_{j_1}^{i-1} \oplus \cdots \oplus s_{j_m}^{i-1} \oplus s_{n-1}^{i-1} \oplus s_{u_1}^{i-1} s_{u_2}^{i-1} \oplus K_{(i-1) \mod |K|}.$$

and

$$s_j^i = \begin{cases} s_{j-1}^{i-1}, & \text{for } 1 \leq j \leq (n-1) \\ f^i, & \text{for } j = 0 \end{cases}$$

To find the differential trails for such ciphers $\mathcal{C}$,$\mathcal{C}'$, the probability is only paid for the active AND gates through rounds. Thus, given an l round differential trail, the overall probability can be calculated by counting only the total number of active ANDs in the trail. Also, it is to be noted that, the whole state bits become unknown after n number of rounds. In another way, we can say that exactly $n - i$ number of state bits are still known for the initial i $(1 \leq i \leq n)$ rounds. Therefore, in chosen plaintext scenario, we can deterministically bypass some of the active AND gates by fixing the message bits for up to some initial i $(\leq n)$ rounds. This characteristic of any NLFSR-based ciphers $\mathcal{C}$,$\mathcal{C}'$ is described in the following lemma.

Lemma 2. *For a cipher $\mathcal{C}'$, forward differential trail for the first $(u_1 + 1)$ rounds is completely free. For the next $(u_2 - u_1)$ rounds, if the input differential to the AND gate is 0 and 1 (i.e., $\Delta s_{u_1} = 0, \Delta s_{u_2} = 1$) then the output of the AND gate can be determined with probability 1 (**conditionally free**). Similarly, for a cipher $\mathcal{C}$, $(n - u_2)$ rounds are completely free and $(u_2 - u_1)$ rounds are conditionally free.*

Proof. As both the inputs to AND gate are known for the first $(u_1 + 1)$ rounds, the output difference of the AND gate can be bypassed with probability 1. For the next $(u_2 - u_1)$ rounds, the u_1-th bit in the state, i.e., s_{u_1} is still known to us from the given input message. Therefore, at the intermediate rounds i $(u_1 + 1 < i \leq u_2)$ if the input difference corresponding to the AND gate becomes $(0, 1)$, i.e., $\Delta s_{u_1} = 0$ and $\Delta s_{u_2} = 1$, then by Observation 2 the output difference of the AND gate can be deterministically bypassed. The proof for the cipher $\mathcal{C}$ follows a similar approach. □

Note that, in the chosen plaintext attack model (CPA), Lemma 2 can be exploited by carefully choosing the message bits. This, in turn, reduces the degrees of freedom of the message space. Whereas the known plaintext attack model (KPA) helps in discarding some of the message pairs which do not follow a given differential trail.

4.1.2 Computing Backward Differential

While computing the backward differential for a cipher $\mathcal{C}$, the feedback function remains almost the same except only the index of the bits are changed. Consider that the initial state is t^0 and the intermediate state after the i^{th} round is t^i. Then the feedback bit, f^i for the i^{th} round is computed in the following way:

$$f^i \leftarrow t^{i-1}_{j_1-1} \oplus \cdots \oplus t^{i-1}_{j_m-1} \oplus t^{i-1}_{n-1} \oplus t^{i-1}_{u_1-1} t^{i-1}_{u_2-1} \oplus K_{(i-1) \mod |K|}$$

and the state bits are updated as follows:

$$t^i_j = \begin{cases} t^{i-1}_{j-1}, & \text{for } 1 \leq j \leq (n-1) \\ f^i, & \text{for } j = 0. \end{cases}$$

Similarly, for cipher $\mathcal{C}'$, the feedback bit is computed as

$$f^i = t^{i-1}_{j_1+1} \oplus \cdots \oplus t^{i-1}_{j_m+1} \oplus t^{i-1}_{0} \oplus t^{i-1}_{u_1+1} t^{i-1}_{u_2+1} \oplus K_{(i-1) \mod |K|}$$

and

$$t^i_j = \begin{cases} t^{i-1}_{j+1}, & \text{for } 0 \leq j \leq (n-2) \\ f^i, & \text{for } j = n-1. \end{cases}$$

Lemma 3. *For a cipher $\mathcal{C}'$, backward differential trail for first $(n-u_2-1)$ rounds is completely free. The next (u_2-u_1) rounds are conditinally free. Similarly, for a cipher $\mathcal{C}$, the first (u_1-1) rounds are completely free whereas the next (u_2-u_1) rounds are conditionally free.*

Proof. The proof is quite similar to that of Lemma 2 □

4.2 NLFSR-based Ciphers with Multiple AND Gates in the Feedback Function

Consider an n-bit NLFSR-based block cipher $\mathcal{D}$ with the initial state value as $s^0 = (s^0_0, s^0_1, \cdots, s^0_{n-1})$. At each round i, the feedback bit f^i is computed in the following way.

$$f^i \leftarrow s^{i-1}_{j_1} \oplus \cdots \oplus s^{i-1}_{j_m} \oplus s^{i-1}_{n-1} \oplus s^{i-1}_{u_1} s^{i-1}_{v_1} \oplus \cdots \oplus s^{i-1}_{u_h} s^{i-1}_{v_h} \oplus K_{i-1},$$

where

1. k^{i-1} is the key bit used in the i^{th} round,
2. $j_1, \cdots, j_m, n-1$ are the taps of the NLFSR,
3. u_j, v_j are the inputs to the AND gate A_j such that $u_j < v_j \leq n-1$, $1 \leq j \leq h$,
4. $j_1 < j_2 \implies u_{j_1} < u_{j_2}$.

Also, the state in the next round is updated in the following way.

$$s^i_j = \begin{cases} s^{i-1}_{j-1}, & \text{for } 1 \leq j \leq (n-1) \\ f^i, & \text{for } j = 0. \end{cases}$$

Lemma 4. *For a cipher $\mathcal{D}$, in the forward differential, the output of gate A_j is deterministic for the first (u_j+1) rounds. For the next (v_j-u_j) rounds, the output of the AND gate is conditionally free. Similarly, for a cipher $\mathcal{D}'$, the* reverse-feed *cipher of $\mathcal{D}$, the output of gate A_j is deterministic for the first $(n-v_j)$ rounds and conditionally free for the next (v_j-u_j) rounds.*

Proof. For cipher $\mathcal{D}$, as $s^i_{u_j}$ and $s^i_{v_j}$ are known for $0 \leq i \leq u_j$, so ΔA_j can be deterministically computed for the first $(u_j + 1)$ number of rounds as both inputs to the `AND` gate are known.

Suppose, during the intermediate rounds, $s^i_{v_j}$ is known and $s^i_{u_j}$ is unknown for $u_j + 1 \leq i \leq v_j$ (round number $u_j + 2$ to $v_j + 1$). If $\Delta s^i_{v_j} = 0$ and $\Delta s^i_{u_j} = 1$, then by Observation 3, $\Delta A_j = s^i_{v_j}$. Hence, for round $u_j + 1$ to v_j, ΔA_j can be determined with probability 1 when such conditions are met.

For cipher $\mathcal{D}'$, $s^i_{u_j}$ and $s^i_{v_j}$ are known for $0 \leq i \leq (n - v_j - 1)$. Hence, ΔA_j can be determined completely free for first $(n - v_j)$ rounds. $s^i_{u_j}$ is known and $s^i_{v_j}$ is unknown for $(n - v_j) \leq i \leq (n - u_j - 1)$ (round number $(n - v_j + 1)$ to $(n - u_j)$). If $\Delta s^i_{v_j} = 1$ and $\Delta s^i_{u_j} = 0$, then by Observation 2, $\Delta A_j = s^i_{u_j}$. Therefore, for next $(v_j - u_j)$ rounds, ΔA_j can be determined with probability 1 when such conditions are met. □

In the same fashion, computing the backward differential, the feedback bit f^i for i^{th} round is computed as

$$f^i \leftarrow t^{i-1}_{l_1+1} \oplus \cdots \oplus t^{i-1}_{l_m+1} \oplus t^{i-1}_0 \oplus t^{i-1}_{u_1+1} t^{i-1}_{v_1+1} \oplus \cdots \oplus t^{i-1}_{u_h+1} t^{i-1}_{v_h+1} \oplus k'^{i-1}$$

and the state in the next round is updated as

$$t^i_j = \begin{cases} t^{i-1}_{j-1}, & \text{for } 0 \leq j \leq (n-2) \\ f^i, & \text{for } j = n-1. \end{cases}$$

Lemma 5. *For a cipher $\mathcal{D}$, in the backward differential, the output of gate A_j is deterministic for first $(n - v_j - 1)$ rounds. For the next $(v_j - u_j)$ rounds, the output of the gate is conditionally free. Similarly, for a cipher $\mathcal{D}'$, the* reverse-feed *cipher of $\mathcal{D}$, in the backward differential the output of gate A_j is deterministic for first (u_j) rounds and conditionally free for next $(v_j - u_j)$ rounds.*

Proof. For cipher $\mathcal{D}$, as $t^i_{u_j+1}$ and $t^i_{v_j+1}$ are known for $0 \leq i \leq n - v_j - 2$, so ΔA_j can be deterministically computed for first $(n - v_j - 1)$ number of rounds as both inputs to the `AND` gate are known.

$t^i_{v_j+1}$ is known and $t^i_{u_j+1}$ is unknown for $n - v_j - 1 \leq i \leq n - u_j - 2$ (round number $n - v_j$ to $n - u_j - 1$). If $\Delta t^i_{u_j+1} = 0$ and $\Delta t^i_{v_j} = 1$, then by Observation 2, $\Delta A_j = t^i_{u_j+1}$. Hence, for round $n - v_j$ to $n - u_j - 1$, ΔA_j can be determined with probability 1 when such conditions are met.

In similar way, it can be proved for $\mathcal{D}'$. □

4.3 Generalization of Chained ANDs

Consider an n-bit cipher $\mathcal{C}$ with $(s_{u_1}, s_{u_2}), (s_{u_2}, s_{u_3})$ and $(\Delta s_{u_1} = 1, \Delta s_{u_2} = 0), (\Delta s_{u_2} = 0, \Delta s_{u_3} = 1)$ are respectively two sequential inputs and their differences to the `AND` gate. Suppose we have differential trail and at the round i, we see that the input diiference $\Delta s_{u_1} = 1, \Delta s_{u_2} = 0$ happens at the `AND` gate and Δz be the coresponding output difference. Then, according to Observation 3, the internal state bit s_{u_2} will be revealed due to the relation $\Delta z = s_{u_2}$. Thus, after the $(u_2 - u_1 - 1)$ number of rounds, i.e., at the round $i + (u_2 - u_1 - 1)$, $\Delta s_{u_2} = 0, \Delta s_{u_3} = 1$ becomes the input difference to the `AND` gate. In this case, by Observation 2, this active `AND` gate will be freely bypassed as we know the bit value s_{u_2}. Therefore, if the subsequent input differences to the `AND` gate are $1, 0, 1$ then instead of paying the probability of $\frac{1}{4}$, we only have to pay the probability of $\frac{1}{2}$. In another way, we can say that when this subsequent $1, 0, 1$ bit difference arise in the `AND` gate, we will count it as one active `AND`. Because, out of two subsequent active `AND`s, we only pay the probability for the first one (i.e., when $\Delta s_{u_1} = 1, \Delta s_{u_2} = 0$) whereas the second (where $\Delta s_{u_2} = 0, \Delta s_{u_3} = 1$) one will pass with probability 1.

Table 3: An Example of ABP

Round	NLFSR State Bit Positions					
	Δs_{12}	Δs_{10}	Δs_8	Δs_5	Δs_3	Δs_1
i	0	0	1	1	0	1
$i+5$	0	1	0	0	0	0
$i+7$	1	0	1	0	0	0
$i+9$	0	1	0	0	0	0

In the previous work [8], for TinyJAMBU cipher, they added some extra constraints in the simple MILP model and recorded all the two subsequent ANDs with $1, 0, 1$ bit differences which helps to increase the overall probability of the differential trail. We named this kind of two subsequent ANDs with $1, 0, 1$ bit differences as Chained AND Bit Pattern (CABP). Now, when shifted to NLFSR with multiple ANDs-based cipher, then there might arise more than two subsequent ANDs with various bit difference patterns which signficantly increase the overall probability of the trail and we named it as AND Bit Pattern (ABP). Before going to define it, we give one example to show how ABP increase the probability in the trail.

Suppose, we have an n-bit cipher $\mathcal{D}$ with two ANDs, where $n = 32$ and $(3, 8), (10, 12)$ are the two different AND's input positions in the NLFSR state. At the round i (> 12), we assume that a particular bit difference $\Delta s_8 = 1, \Delta s_5 = 1, \Delta s_3 = 0, \Delta s_1 = 1$ happens in the state. Also, we choose the bit difference 0 at the third position in the state as a pivot. In the subsequent rounds, this pivot will active the ANDs in the following way.

1. At round i, since $\Delta s_8 = 1, \Delta s_3 = 0$ happens, we will first recover the state bit at the pivotal position according to Observation 2.
2. Then, at the round $i+7$, the pivot goes to the bit position 10 and activate the second AND gate as $\Delta s_{12} = 1, \Delta s_{10} = 0$. Thus, according to the Observation 3, this active AND will be freely passed.
3. Similarly, when the pivot goes to the 12-th position in the state at the round $i+9$, the AND will be passed detrministically according to the Observation 2.

The above steps are summarized in the Table 3. In this example, we have to only pay the probability of 2^{-1} instead of 2^{-3}, as the total number of active ANDs subject to the pivot is 3.

Definition 1. AND Bit Pattern (ABP). Consider the cipher $\mathcal{D}$. The ABP of a pivotal bit difference $\Delta s^i_j = 0$ (i is the round number and j is the bit position) is denoted by $\alpha_{\mathcal{D}}(i, j)$ and is defined as a $(2h + 1)$-bit string in the following way:

$$\alpha_{\mathcal{D}}(i, j) = l_h\, l_{h-1}\, \cdots\, l_1\, \Delta s^i_j\, r_1\, \cdots\, r_{h-1}\, r_h.$$

Where l_p= bit difference at position u_p when Δs^i_j shifts to position v_p after some rounds and r_p= bit difference at position v_p when Δs^i_j shifts to position u_p where $1 \le p \le h$. When $\exists$ at least one $p \in \{1, \cdots, h\}$ such that $l_p = r_p = 1$, then $\alpha_{\mathcal{D}}(i, j)$ is a special case of ABP which we called as Chained AND Bit Pattern (CABP).

Definition 2. For a cipher $\mathcal{D}$, $\Omega^{A_j}_{\Delta b}$ is defined as the output difference of AND gate A_j when the bit difference Δb shifts to position s_{v_j}.

Definition 3. For cipher $\mathcal{D}$, $\Theta^{A}_{\Delta b}$ is defined as the output difference of AND gate A_j when the bit difference Δb shifts to position s_{u_j}.

Definition 4. The weight of a `CABP` $\alpha_{\mathcal{D}}(i,j)$ is denoted by $wt(\alpha_{\mathcal{D}}(i,j))$ and is defined as the number of 1's in the `CABP`.

Lemma 6. *Consider a `CABP` with $\alpha_{\mathcal{D}}(i,j) = l_h \,\cdots\, l_1 \,\Delta s_j^i\, r_1 \,\cdots\, r_h$ and $wt(\alpha_{\mathcal{D}}(i,j)) = p+q$. If $l_{w_1} = \cdots = l_{w_p} = r_{y_1} = \cdots r_{y_q} = 1$ then $\Omega_{\Delta s_j^i}^{A_{w_1}} = \cdots = \Omega_{\Delta s_j^i}^{A_{w_p}} = \Theta_{\Delta s_j^i}^{A_{y_1}} = \cdots = \Theta_{\Delta s_j^i}^{A_{y_q}}$.*

Proof. According to the Observation 3, if $l_{w_g} = 1$ and $\Delta s_j^i = 0$ then $\Omega_{\Delta s_j^i}^{A_{w_g}} = s_j^i$ holds $\forall$ $g \in \{1, \cdots, p\}$. Similarly, as $r_{y_g} = 1$ and $\Delta s_j^i = 0$, $\Theta_{\Delta s_j^i}^{A_{w_g}} = s_j^i$ holds where $1 \le g \le q$. Hence, we can conclude that $\Omega_{\Delta s_j^i}^{A_{w_1}} = \cdots = \Omega_{\Delta s_j^i}^{A_{w_p}} = \Theta_{\Delta s_j^i}^{A_{y_1}} = \cdots \Theta_{\Delta s_j^i}^{A_{y_q}}$. □

Lemma 7. *Let $wt(\alpha_{\mathcal{D}}(i,j)) = m$ and $m \ge 2$. Then the subsequent output differences of m active `AND` gates can be restricted to probability 2^{-1} instead of 2^{-m}.*

Proof. As $wt(\alpha_{\mathcal{D}}(i,j)) = m$, then from Lemma 6 it can be conclude that output of m `AND` gates should be s_j^i. Therefore, the value of s_j^i can be fixed with probability 2^{-1}. □

Example 2. Consider the NLFSR-based block cipher `TinyJAMBU` which has one `AND` gate in the feedback function. Its `CABP` should be of the form $l_1 0 r_1$. When $l_1 = r_1 = 1$, then by Lemma 7 output difference of two `AND` gates are fixed with probability 2^{-1} instead of 2^{-2}. This is reported in the refined model [8].

4.4 MILP Modeling of CABP

The number of valid patterns of `CABP` which captures the dependency among the output differences of subsequent active `AND` gates can is described in the following Lemma 8.

Lemma 8. *The total number of valid patterns of `CABP` $\alpha_{\mathcal{D}}(i,j)$ will be $\lambda\; (= \sum_{m=2}^{2h} \binom{2h}{m})$.*

Proof. Consider an `ABP` with $wt(\alpha_{\mathcal{D}}(i,j)) = m$. There are $\binom{2h}{m}$ valid patterns of $\alpha_{\mathcal{D}}(i,j)$ which shows the dependency between m subsequent active `AND` gates.

By Lemma 7, for a `CABP`, if $m \ge 2$, then we have shown a dependency between the output differences of `AND` gates. Therefore, the total number of valid `CABP` will be $\binom{2h}{2} + \binom{2h}{3} + \cdots \binom{2h}{2h}$. □

For modeling the dependency among the subsequent active `AND` gates, the approach is quite similar to the model given in [8]. To do so, first, a constraint is used to identify which `AND` gates are correlated and then pairs of `AND` gates are considered to model the dependency between them. So, to capture any bit difference pattern in the `CABP` with $m \ge 2$, we have added some extra constraints corresponding to the chained active `AND` gates in the simple MILP modeling. To model $\mathcal{D}$, we assign $j = u_1$ to fix the pivot position in the `CABP` $\alpha_{\mathcal{D}}(i,j)$. As the `CABP` $\alpha_{\mathcal{D}}(i,u_1)$ has λ different valid patterns, we take γ_z, $1 \le z \le \lambda$ to capture the correlation among $wt(\alpha_{\mathcal{D}}(i,u_1))$ number of active `AND` gates.

Thus for all the pivot postions at u_1 in the consecutive rounds of the state, the following constraints will be added. For any `CABP` with $wt(\alpha_{\mathcal{D}}(i,u_1)) = p+q$, we compute γ_z in the following way.

$$\gamma_z = l_{w_1} \cdots l_{w_p} \bar{l}_{w_1'} \cdots \bar{l}_{w_{p'}'} \bar{\Delta} s_j^i r_{y_1} \cdots r_{y_q} \bar{r}_{y_1'} \cdots \bar{r}_{y_{q'}'}$$

Where, $l_{w_1} = \cdots = l_{w_p} = r_{y_1} = \cdots r_{y_q} = 1$ and $l_{w_1'} = \cdots = l_{w_{p'}'} = r_{y_1'} = \cdots r_{y_{q'}'} = 0$ such that $\{w_1, \cdots, w_p\} \cup \{w_1^{'}, \cdots, w_{p'}^{'}\} = \{u_1, \cdots, u_h\}, \{w_1, \cdots, w_p\} \cap \{w_1^{'}, \cdots, w_{p'}^{'}\} = \emptyset, \{y_1, \cdots, y_q\} \cup \{y_1^{'}, \cdots, y_{q'}^{'}\} = \{v_1, \cdots, v_h\}, \{y_1, \cdots, y_q\} \cap \{y_1^{'}, \cdots, y_{q'}^{'}\} = \emptyset$.

According to Lemma 6, we have $\Omega_{\Delta s_j^i}^{A_{w_1}} = \cdots = \Omega_{\Delta s_j^i}^{A_{w_p}} = \Theta_{\Delta s_j^i}^{A_{y_1}} = \cdots \Theta_{\Delta s_j^i}^{A_{y_q}}$. Therefore, for each of λ valid bit difference patterns of `CABP`, the following constraints are constructed to capture its correlation.

Table 4: Best Type 4 Trails of TinyJAMBU Correspond to Different MILP Models. ? denotes that the solver has not stopped.

Rounds	**Simple Model** [1]	**Refined Model** [8]	DEEPAND **Model**
192	4	4	2
320	13	12	8
384	–	19	14
480	–	29?	22
640	–	53?	42
1024	–	–	108?

1. $\Omega_{\Delta s_j^i}^{A_{w_{i'}}} - \Omega_{\Delta s_j^i}^{A_{w_{j'}}} \leq 1 - \gamma_z$
   $\Omega_{\Delta s_j^i}^{A_{w_{j'}}} - \Omega_{\Delta s_j^i}^{A_{w_{i'}}} \leq 1 - \gamma_z$, $1 \leq i' < j' \leq p$

2. $\Theta_{\Delta s_j^i}^{A_{y_{i'}}} - \Theta_{\Delta s_j^i}^{A_{y_{j'}}} \leq 1 - \gamma_z$
   $\Theta_{\Delta s_j^i}^{A_{y_{j'}}} - \Theta_{\Delta s_j^i}^{A_{y_{i'}}} \leq 1 - \gamma_z$, $1 \leq i' < j' \leq q$

3. $\Omega_{\Delta s_j^i}^{A_{w_{i'}}} - \Theta_{\Delta s_j^i}^{A_{y_{j'}}} \leq 1 - \gamma_z$
   $\Theta_{\Delta s_j^i}^{A_{y_{j'}}} - \Omega_{\Delta s_j^i}^{A_{w_{i'}}} \leq 1 - \gamma_z$, $1 \leq i' \leq p$, $1 \leq j' \leq q$

Now, this model, which we called as DEEPAND model, is applied to find the good differentials for KATAN and TinyJAMBU.

5 Attacks on TinyJAMBU

The refined model developed in this paper are applied to mount attacks on variants of *keyed* permutation $\mathcal{P}_1$ of TinyJAMBU. First of all, the previous results are discussed and then the results of this paper are elaborated.

5.1 Summary of Previous Results

The designers of TinyJAMBU have specified four different constraints regarding the input-output active-bit positions of $\mathcal{P}_1$ while searching for its differential trail [1]. However, in the context of this work the condition where no constraint is imposed on the input and output of $\mathcal{P}_1$ (Type 4) is relevant and thus similar results are discussed here.

The designers have considered differential trails where each AND gates are treated independently (simple model). Later on, Saha *et al.* reported differential trails where the probability of differential trails are improved by considering the correlation between two different AND gates [8]. This new model is called as refined model. For 320 rounds of $\mathcal{P}_1$, the maximum probability in the simple model is 2^{-13}, whereas in the refined model the probability for the same number of rounds is increased to 2^{-12}. Table 4 compares the minimum number of effectively active AND gates for various number of rounds of $\mathcal{P}_1$.

Table 5: Type 4 Differential Trails of $\mathcal{P}_{\mathbf{384}}$ with Probability 2^{-14}

Input:	$\Delta S_{127\cdots 0}$	`0x00000000`	`0x88040000`	`0x00000248`	`0x02000043`
	$\Delta S_{255\cdots 128}$	`0x00000000`	`0x80000000`	`0x00010000`	`0x00000012`
	$\Delta S_{383\cdots 256}$	`0x00000000`	`0x80000000`	`0x00000000`	`0x00000000`
Output:	$\Delta S_{511\cdots 384}$	`0x04080000`	`0x80004000`	`0x00010200`	`0x00000010`

Table 6: Part of differential trail of `TinyJAMBU` showing the effect of Observation 2.

#Rnd	$\Delta s_{70\cdots 85}$
42	0000000000000000
43	0000000000000000
44	0000000000000000
45	0000000000000000
46	0000000000000000
47	0000000000000000
48	0000000000000000
49	0000000000000001
50	0000000000000010
51	0000000000000100
52	0000000000001001
53	0000000000010010
54	0000000000100100
55	0000000001001000
56	0000000010010000
57	0000000100100000

5.2 MILP Modeling for Finding Differential Trail

As the design of `TinyJAMBU` is similar to the cipher described in Section 4.1, from Lemma 2 it can be concluded that the first (128-85)=43 rounds are completely free and the next (85-70)=15 rounds are conditionally free. For the rest number of rounds refined modeling [8] is employed.

5.3 Attacks on $\mathcal{P}_1$

To find the differential characteristics of $\mathcal{P}_1$, in addition to the refined model, the Observation 1 and Observation 2 are employed to improve the probability. By Lemma 2 it can be concluded that the first (128-85-1)=42 rounds is completely free and the next (85-70)=15 rounds is free whenever difference 0 and 1 appears in the input of the AND gate where the actual bit corresponding to input difference 0 is known.

Refer to Table 6. Consider the bits 70 and 85 in round number 43 to 57 of the trail given in Table 5. It is clearly evident from the table, that in round number 49 and 52, $\Delta s_{70}^i = \Delta s_{70}^i = 0$ and $\Delta s_{85}^i = \Delta s_{85}^i = 1$. As s_{70}^i and s_{70}^i is known, the output differential of the corresponding `AND` gate is deterministic. Hence, this gives a factor of 2^2 advantage in the probability.

Cluster Differential Trail for 384 Rounds

By employing the above refined model in MILP, we are able to find differential trails with more improved probabilities. For 320 rounds, our model gives a differential trail with probability 2^{-8} which is much better than previously reported results. For $\mathcal{P}_{\mathbf{384}}$, a differential trail with probability 2^{-14} is found. The trail is shown in Table 5. We obtained 4 differential trails with the same input and output difference as shown in Table 5 each with probability 2^{-14}, 2^{-15}, 2^{-16} and 2^{-17}. Thus the overall probability for the differential trail is $2^{-13.17}$.

Differential Trail of $\mathcal{P}_{\mathbf{640}}, \mathcal{P}_{\mathbf{1024}}$

In `TinyJAMBU` v2, the designers have increased the number of rounds of the permutation which is used to process nonce and associated data to 640 rounds [1, Page 12] due to effect of correlated AND gates as shown in [8]. The MILP model developed in this work is applied on the keyed permutations $\mathcal{P}_{\mathbf{640}}$ and $\mathcal{P}_{\mathbf{1024}}$. For, $\mathcal{P}_{\mathbf{640}}$, we have found a trail with probability 2^{-42}. However, for $\mathcal{P}_{\mathbf{1024}}$, as the solver is unable to finish due to a higher number of rounds, we have found upto a differential trail with probabilty 2^{-108}.

6 Attacks on `KATAN`

Here, we apply the MILP model developed in this paper to mount attacks on `KATAN` . First, we show that for certain number of initial rounds for variants of `KATAN` , differential characteristics with much better probability, in comparison to the designer's claims [5], can be found. Then we have shown that the related key boomerang attacks against `KATAN` [6] can also be improved by employing the same model.

6.1 MILP Model for Differential Cryptanalysis of `KATAN`

Modeling the Free Rounds.

Consider that in the generalized design of `KATAN` , the bits y_3, y_4 ($y_3 > y_4$) are inputs to `AND` gate A_1, y_5, y_6 ($y_5 > y_6$) are inputs to gate A_2 and x_3, x_4 ($x_3 > x_4$) are inputs to gate A_3. Then by Lemma 4, the differential output of gate A_1, A_2 and A_3 in the forward differential trail are deterministic for first $(y_4 + 1)$ rounds, first $(y_6 + 1)$ rounds and first $(x_4 + 1)$ rounds respectively and conditionally free from round number $(y_4 + 2)$ to $(y_3 + 1)$, $(y_6 + 2)$ to $(y_5 + 1)$ and $(x_4 + 2)$ to $(x_3 + 1)$ respectively.

Similarly, by Lemma 5, it can be concluded that in the backward differential trail, the differential output of gate A_1, A_2 and A_3 are deterministic for the first $(n - y_3 - 1)$ rounds, first $(n - y_5 - 1)$ rounds and first $(n - x_3 - 1)$ rounds respectively and conditionally free from round number $(n - y_3)$ to $(n - y_4 - 1)$, $(n - y_5)$ to $(n - y_6 - 1)$ and $(n - x_3)$ to $(n - x_4 - 1)$ respectively.

Example 3. For `KATAN32` , in the forward differential trail, the differential output of gate A_1 is completely free for the first (10+1)=11 rounds and is conditionally free from round number (10+2)=12 to (12+1)=13. Table 7 shows the number of completely free rounds and conditionally free rounds for variants of `KATAN` .

Modeling the Dependency Between `AND` Gates

In the L_1 register, there is only one `AND` gate. Thus the refined modeling described in [8] can be employed to find the differential trails. In L_2 register, there are two `AND` gates and the dependency between two different `AND` gates is not captured in the refined model. Consider a bit s_3^i in register L_2. For `KATAN32` , as $(y_5 - y_6) > (y_3 - y_4)$, the ABP of Δs_3^i is $\Delta s_8^i \Delta s_{12}^{i+7} \Delta s_3^i \Delta s_{10}^{i+9} \Delta s_3^{i+5}$ which can also be considered as $\Delta s_8^i \Delta s_5^i \Delta s_3^i \Delta s_1^i \Delta s_{-2}^i$ $(\Delta s_j^{i+k} = \Delta s_{j-k}^i)$ where $\Delta s_{-k}^i = \Delta s_0^{i+k}$.

Table 7: Number of completely free rounds and conditionally free rounds for `AND` gate A_1, A_2 and A_3. r_{free} denotes the round numbers in which the differential output of the `AND` gate is completely free whereas r_{cond} denotes the round numbers in which the output is conditionally free.

Variant	A_1		A_2		A_3	
	r_{free}	r_{cond}	r_{free}	r_{cond}	r_{free}	r_{cond}
`KATAN32`	$1 \rightarrow 11$	$12 \rightarrow 13$	$1 \rightarrow 4$	$5 \rightarrow 9$	$1 \rightarrow 6$	$7 \rightarrow 9$
`KATAN48`	$1 \rightarrow 14$	$15 \rightarrow 22$	$1 \rightarrow 7$	$8 \rightarrow 16$	$1 \rightarrow 8$	$9 \rightarrow 16$
`KATAN64`	$1 \rightarrow 22$	$23 \rightarrow 34$	$1 \rightarrow 10$	$11 \rightarrow 15$	$1 \rightarrow 12$	$13 \rightarrow 21$

Table 8: CABP of s_3^i and the corresponding differential value of related bits.

CABP	Δs_8^i	Δs_5^i	Δs_1^i	Δs_{-2}^i
11011	1	1	1	1
11010	1	1	1	0
11001	1	1	0	1
10011	1	0	1	1
01011	0	1	1	1
11000	1	1	0	0
10010	1	0	1	0
01010	0	1	1	0
10001	1	0	0	1
01001	0	1	0	1
00011	0	0	1	1

If the CABP is considered, then by Lemma 8 there are $\binom{4}{4} + \binom{4}{3} + \binom{4}{2} = 11$ patterns for which output differential of several `AND` computations are inter-related. The CABP and the corresponding differential bits are shown in Table 8

In `KATAN48` , $y_3 - y_4 > y_5 - y_6$. Consider a difference bit Δs_{21}^i in L_2 register.

$$\begin{aligned}\alpha_{\texttt{KATAN48}}(i,21) &= \Delta s_{33}^i \Delta s_{14}^{i-12} \Delta s_{21}^i \Delta s_9^{i-7} \Delta s_{21}^{i+12} \\ &= \Delta s_{33}^i \Delta s_{26}^i \Delta s_{21}^i \Delta s_{16}^i \Delta s_9^i\end{aligned}$$

6.2 Differential Trails of `KATAN`

In [5], the designers have claimed that for 42-round `KATAN32` , the best differential characteristic has probability 2^{-11}. However, for the initial 42 rounds our MILP model is able to find two *identical* differential trails with probability 2^{-7}. The trail is shown in Fig. 3. It can be observed that in the trail, that $\Delta L_2^8[3] = 1$ and $\Delta L_2^8[8] = 0$. As $L_2^8[8] = L_2^0[0]$, then the corresponding output differential of the `AND` gate A_1 in round 8 is $L_2^0[0]$. Hence, the probability is improved by a factor of 2.

However, one of these trail is invalid which can be figured out by considering the dependency between the AND gates for the initial rounds (the dependency between the AND gates when both the input of the AND gate are known is not considered in our model). Consider,

$$\Delta z_{i,j}^k = L_2^k[i] L_2^k[j] \oplus (L_2^k[i] \oplus \Delta L_2^k[i])(L_2^k[j] \oplus \Delta L_2^k[j])$$

. As, in the trail, $\Delta L_2^2[3] = 0$, $\Delta L_2^2[8] = 1$, $\Delta L_2^2[10] = 0$, $\Delta L_2^2[12] = 1$, $\Delta L_2^2[18] = 1$ and $\Delta L_1^3[0] = 0$, then there are two cases.

#R	$\Delta L_2[0\cdots18]$																			$\Delta L_1[0\cdots12]$												
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	1	0	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	0
10	0	0	0	0	0	1	0	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
11	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	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	0	0	0	0	0	0	0
15	0	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	0	0	0	0	0	0
16	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	0	0	0	0	0	0	0
17	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	0	0	0	0	0	0
18	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	0	0	0	0	0
19	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	0	0	0	0
20	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	0	0	0
21	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	0	0
22	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	0
23	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
24	0	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
25	0	0	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
26	0	0	0	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
27	0	0	0	0	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
28	0	0	0	0	0	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
29	0	0	0	0	0	0	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
30	0	0	0	0	0	0	0	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
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
32	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
33	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
34	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
35	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
36	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
37	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 3: Differential trail for 42-round KATAN32 with probability 2^{-7}. $\#R$ denotes the number of rounds. The differential bits input to AND gates are colored red whereas the bits that are XOR-ed in the feedback function are colored blue.

- Case 1: $\Delta z^2_{3,8} = 0$ and $\Delta z^2_{10,12} = 1$,
- Case 2: $\Delta z^2_{3,8} = 1$ and $\Delta z^2_{10,12} = 0$.

Consider the case 1. As $\Delta z^2_{10,12} = 1$, then by Observation 2, $L^2_2[10] = 1$. Due to the shifting of bits, $\Delta L^4_2[12] = \Delta L^2_2[10] = 0$ and $L^4_2[12] = L^2_2[10]$. As $\Delta L^4_2[10] = 1$, then by Observation 3, $\Delta z^4_{10,12} = \Delta L^4_2[12] = 1$. As all other differential tapping bits are 0, so $\Delta L^5_1[0] = \Delta z^4_{10,12} = 1$. However, it can be observed from Fig. 3 that $\Delta L^5_1[0] = 0$. Hence, this is an invalid trail. It can be verified that case 2 gives a valid trail.

Note that, such invalid trails can only be yielded from the model for some initial number of rounds when both the inputs to the AND gate are known because the constraints regarding the correlation between two AND gates are relaxed. However, such constraints are not relaxed for later number of rounds when both the inputs to the AND gate are not known. Hence, for later number of rounds such invalid trails can not be yielded from the model.

For 43-round KATAN48 and 37-round KATAN64 , the best differential trail, as claimed by the designers, can be found with probability 2^{-18} and 2^{-20} respectively whereas for both variants our model finds differential trails with probability 2^{-14}.

Dependency between Multiple AND Gate

Although, in the optimal trail the DEEPAND model is unable to find dependencies between multiple AND gate, in non-optimal trail it succesfully detects such dependencies. One such trail returned by DEEPAND model is shown in Fig. 4. In the trail, $\Delta L^{15}_2[3] = 0$ and $\Delta L^{15}_2[8] = 1$. So, the corresponding AND gate is active in round 15. Also, $\Delta L^{22}_2[10] = 0$ and $\Delta L^{22}_2[12] = 1$, which also activates the corresponding AND gate. However, the input $\Delta L^{22}_2[10]$ and $\Delta L^{15}_2[3]$ is the same. Thus the above two AND gates are correlated and instead of 2^{-2} and 2^{-1} probability is required to be paid. This is captured by DEEPAND model.

6.3 Related Key Differential Attack

In the related-key setting, we have applied our new model to KATAN 32, and found the best optimal differential trail with probability 2^{-4}. Also, we have found a probabilty of 2^{-6} and 2^{-8} for 75 and 80 rounds respectively.

6.3.1 Related Key Boomerang Attack in [6]

Related key boomerang distinguishers are consists of two different differential trails. The strategy to find differential trails is consists of three steps-

- Collision step
- Blank step
- Brute force step

Initially a plaintext difference and a key difference is provided. At the end of the collision step, it is expected that the both the differences will cancel each other. Thereafter, for some rounds the state difference and the subkey difference remains zero and thus no probability is required to pay for transitions through this rounds. This step is called blank step. Finally, after some rounds the subkey differences introduce some differences in the state and from here the best differential trail is considered. This is the brute force step.

Attack on KATAN32 . It is observed in [6], that carefully choosing the plaintext bits can result in a 49-round blank step and thus it is expected that the overall probability would increase for such distinguishers.

#R	$\Delta L_2[0\cdots 18]$																			$\Delta L_1[0\cdots 12]$												
0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0
1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	0	1	1	0	1	0	0
2	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	0	0	1	0	1	0
3	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	1	1	1	0	1	0	1
4	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1	0
5	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0	1
6	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
7	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
8	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
9	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
22	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
23	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
24	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
25	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0
26	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
27	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0
28	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
29	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
31	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
32	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
33	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
34	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0
35	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0
36	0	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0
37	1	0	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0
38	0	1	0	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
39	0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	1	0	1	1	0	0	1	0
40	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	1	0	0	1	0	0	1
41	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	0	0

Figure 4: Differential trail for 42-round `KATAN32` with probability 2^{-32}. $\#R$ denotes the number of rounds. The differential bits input to AND gates are colored red whereas the bits that are XOR-ed in the feedback function are colored blue.

In [6], 11 input sets based on the initial key differences are defined. For set 8, the upper trail and the lower trail of the distinguisher is constructed where both trails are of 70 rounds each. It is shown that the upper trail has 8 trails with probability 2^{-9}, 16 trails each with probability 2^{-10} and 2^{-11} and 64 trails with probability 2^{-12}. All these trails have same input differences. Thus the cumulative probability is approximately $2^{-7.1}$. Similarly for lower trail the cumulative probability is $2^{-6.5}$. Hence, 140-round related key boomerang distinguisher for KATAN32 is constructed with probability $(2^{-7.1})^2(2^{-6.5})^2 = 2^{-27.2}$.

The DEEPAND MILP model for KATAN32 is devised for finding trail E_0 of [6] and initialized using the set 8. However, our model has not shown any significant improvement over the 70-round related key differential trails found in [6]. One possible reason is for lower number of rounds there is very less chance of observing correlation between AND gates.

7 Conclusion

The DEEPAND model developed in this paper is primarily based on Observation 2 and Observation 3. In this model, it is shown that there can be dependencies in between multiple gates in NLFSR-based ciphers. To capture the dependencies in a proper way, CABP has been introduced. In addition, it is also shown that if one of the inputs of AND gate is known, then for certain values of input differentials of the AND gate, the output differential is deterministic.

In the context of this paper, only the L_2 register in KATAN has more than one AND gate. CABP has been applied for L_2 register. Although the probabilities of the trails have been improved considering only one of the inputs to the AND gates is known, but in the optimal trails that have been furnished in this paper, no dependencies between multiple AND gates have been observed. One possible reason is devising MILP model for a small number of rounds. In case refined modeling of TinyJAMBU [8], it has been observed that for 224 rounds of TinyJAMBU there are no correlated AND gates. However, as the number of rounds is increased, the effect of correlated AND is increased. From this, it can be concluded that in the case of KATAN , if more number of rounds can be penetrated, then the dependencies between multiple AND gate can be observed. However, as of now, we are unable to provide any theoretical bounds on the minimum number of rounds to observe such effects. Another question that requires to be pondered upon is whether the positions of tapping bits have any significant effect on this bound. It would be an interesting open problem to find a relationship between the position of tapping bits, round numbers, and dependencies between multiple AND gates.

The main limitation of this work is due to a large number of constraints in the developed MILP model, the solver does not stop for a large number of rounds. So, in this regard, if an algorithm can be developed which reduces the effective number of constraints, then there is a possibility that the developed model can be applied on more number of rounds, and thus more dependencies between the AND gates can be observed.

References

[1] Hongjun Wu and Tao Huang. TinyJAMBU: A Family of Lightweight Authenticated Encryption Algorithms. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/tinyjambu-spec-final.pdf. NIST LWC Finalist, 2021.

[2] Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M. Youssef. MILP modeling for (large) s-boxes to optimize probability of differential characteristics. *IACR Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.

[3] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. *J. Cryptol.*, 4(1):3–72, 1991.

[4] Christina Boura and Daniel Coggia. Efficient MILP modelings for sboxes and linear layers of SPN ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(3):327–361, 2020.

[5] Christophe De Cannière, Orr Dunkelman, and Miroslav Knežević. Katan and ktantan — a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, pages 272–288, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[6] Takanori Isobe, Yu Sasaki, and Jiageng Chen. Related-Key Boomerang Attacks on KATAN32/48/64. In Colin Boyd and Leonie Simpson, editors, *Information Security and Privacy*, pages 268–285, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[7] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[8] Dhiman Saha, Yu Sasaki, Danping Shi, Ferdinand Sibleyras, Siwei Sun, and Yingjie Zhang. On the Security Margin of TinyJAMBU with Refined Differential and Linear Cryptanalysis. *IACR Transactions on Symmetric Cryptology*, 2020(3):152–174, Sep. 2020.

[9] Siwei Sun, Lei Hu, Ling Song, Yonghong Xie, and Peng Wang. Automatic security evaluation of block ciphers with s-bp structures against related-key differential attacks. In Dongdai Lin, Shouhuai Xu, and Moti Yung, editors, *Information Security and Cryptology - 9th International Conference, Inscrypt 2013, Guangzhou, China, November 27-30, 2013, Revised Selected Papers*, volume 8567 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2013.

[10] Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song, and Kai Fu. Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties. Cryptology ePrint Archive, Paper 2014/747, 2014. https://eprint.iacr.org/2014/747.

[11] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, DES(L) and other bit-oriented block ciphers. In Palash Sarkar and Tetsu Iwata, editors, *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*, volume 8873 of *Lecture Notes in Computer Science*, pages 158–178. Springer, 2014.