Modeling Large S-box in MILP and a (Related-Key) Differential Attack on Full Round PIPO-64/128

Tarun Yadav(✉) and Manoj Kumar

Scientific Analysis Group, DRDO, Metcalfe House Complex, Delhi 110 054, India
{tarunyadav.sag,manojkumar.sag}@gov.in

Abstract. The differential characteristic search problem is converted into mixed integer linear programming (MILP) model to get the bound against differential attack. The difference distribution table is used to write the linear inequalities for MILP modeling of S-box. To construct a reduced set of such inequalities, we present the approaches based on Quine-McCluskey(QM) and Espresso algorithms that are used for active S-box minimization and probability optimization respectively. These approaches are used to search the differential characteristics for lightweight block cipher PIPO-64/128. There are 20621 inequalities in 23 variables corresponding to possible difference transitions in the DDT and these are minimized to 6035 inequalities. MILP model based on these inequalities is used to optimize the probability of differential and impossible differential characteristics for PIPO-64/128 reduced to 9 and 4 rounds respectively. We construct an iterative 2-round related-key differential characteristic with the probability of 2^{-4} and that is used to present a full round related-key differential distinguisher with the probability of 2^{-24}. We develop a key recovery attack using related keys on full round PIPO-64/128 with the data complexity of 2^{32}. We present a major collision in PIPO-64/128 which produces the same ciphertext (C) by encrypting the plaintext (P) under two different keys.

Keywords: Block cipher · Differential cryptanalysis · Lightweight cryptography · MILP · S-box

1 Introduction

Differential attack is one of the most powerful techniques for the cryptanalysis of block ciphers [4]. For new block ciphers, it is a mandatory design criterion to provide proof of resistance against the differential attack [6]. The high probability relations between the input and output differences of a block cipher are utilized to distinguish it from the uniform distribution [5]. We need a differential characteristic with the probability of 2^{-p}, where $p \lll n$, to mount the attack on n-bit block cipher [16]. To estimate the strength of a block cipher against differential attack, we calculate a lower bound on the number of active

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
L. Batina et al. (Eds.): SPACE 2022, LNCS 13783, pp. 3–27, 2022.
https://doi.org/10.1007/978-3-031-22829-2_1

S-boxes in a differential characteristic. Then, an upper bound on the probability is estimated using this lower bound and maximum differential probability of the S-box [18]. Initially, branch-and-bound based techniques were used to search the high probability differential characteristics [19,23]. Nowadays, automated solvers based on mixed integer linear programming (MILP) [24], satisfiability modulo theory (SAT/SMT) [10], constraint programming (CP) problems [13,33], and machine learning based techniques [14,34] are used to test the differential attack resistance. In 2012, MILP-aided differential cryptanalysis for block ciphers was proposed by Mouha *et al.* This technique proved to be very successful to mount the differential attack on block ciphers.

Mixed integer linear programming is used frequently to solve optimization problems. MILP deals with optimizing the objective function $f(x_1, x_2, \cdots, x_n)$ subject to a set of linear inequalities $Ax \leq b$ which involves decision variables $x_i, 1 \leq i \leq n$ with restrictions on certain variables to take integer values. We can convert the differential characteristic search problem into a MILP model [24]. Then, optimization problem solvers (viz. Gurobi [15] and CIPLEX [11]) are used to solve the MILP model to get a lower bound on the number of active S-boxes and search for high probability differential characteristics. The linear layers (viz. key addition and permutation layer) of a block cipher are easily converted into linear inequalities. The S-box is a non-linear component of the block cipher and DDT of the S-box is used to write the linear inequalities satisfying each possible propagation. This set contains a large number of inequalities and it becomes hard to solve the MILP model based on this set. Therefore it is required to minimize the number of inequalities to obtain the solution efficiently. Various methods have been proposed in the literature to optimize the number of inequalities in this set.

Mouha *et al.* showed the use of MILP in differential cryptanalysis of block ciphers and used optimization solvers to get the security bounds [24]. They presented a framework to get the least number of active S-boxes in a differential characteristic of word oriented ciphers. This technique was illustrated on Advanced Encryption Standard (AES) and least number of active S-boxes in 4-round differential characteristic of AES were obtained by solving the MILP model.

Sun *et al.* extended the use of MILP for bit oriented ciphers and two methods based on logical condition modeling and convex hull computation were proposed to get the MILP model of S-box [27,28]. The DDT of S-box was used to write the linear inequalities for possible propagations using the SageMath tool [29]. Then greedy search algorithm was used to reduce the number of inequalities in this model. For a 4-bit S-box, the reduced set contains about 30 inequalities. Due to the limitation of SageMath, this method is not practical for the S-box of size greater than 6-bit. Sasaki and Todo proposed another method for MILP modeling of S-box to reduce the number of inequalities [30–32]. They proposed a MILP based method to reduce the inequalities using impossible propagations in the DDT. For a 4-bit S-box, this method provides around 20 linear inequalities that are used to minimize the number of active S-boxes. This method was also

used to model the MILP problem for lightweight block cipher WARP [20]. This method uses SageMath to write the inequalities, therefore it also does not work for S-boxes of size more than 6-bit.

For 8-bit S-box, 16 variables are needed to write the linear inequalities for possible and impossible difference transitions in the DDT. For large S-boxes, Abdelkhalek *et al.* generated the linear inequalities using Logic Friday [22] (based on Espresso algorithm). The *pb*-DDT approach was proposed to optimize the probability of a differential characteristic by separating the DDT into multiple tables according to the probabilities. Boura and Coggia [8] proposed another approach to generate and minimize the number of linear inequalities for 8-bit S-boxes based on the impossible transitions in the DDT. This method was used to minimize the number of linear inequalities for AES S-box in [8]. The time complexity to get the 2882 linear inequalities for AES S-box was 22 d. They did not mention about the number of linear inequalities for partial or full DDT that will be required to optimize the probability of differential characteristics.

Our Contribution: The existing works focused on minimizing the number of linear inequalities to represent the DDT of large S-boxes. Whereas, the time complexity to minimize the number of inequalities for large S-boxes was several days [8]. Our aim is to generate a minimized set of linear inequalities within the practical time limit ($\leq$5 h). We present a new method to generate the additional set of linear inequalities using intermediate output of the QM algorithm and get the minimized set of linear inequalities in practical time. We also solve the MILP model to optimize the number of active S-boxes in PIPO using two different set of linear inequalities. These experiments show that there is no significant difference in the time complexity to solve the MILP problem using large or small set of linear inequalities. For probability optimization part, we generate the linear inequalities for full DDT using our tool MILES (based on Espresso). We show the application of MILES to search the differential, impossible differential and related-key differential characteristics of lightweight block cipher PIPO-64/128 [17]. We achieve the designer's bound for differential and impossible differential characteristics. We present the full round related-key differential distinguishers and mount a key recovery attack on full round PIPO-64/128. Using MILES and MILP modeling of related-key differential search, we show the collisions in PIPO-64/128.

Organisation: The paper is organised as follows. In Sect. 2, we discuss MILP modeling of block ciphers with 8-bit S-boxes. We present approaches based on QM and Espresso algorithms to minimize the number of linear inequalities and compare the results for AES, SKINNY and PIPO S-boxes. In Sect. 3, we show the application of MILES to model the MILP problem to optimize the probability of differential characteristics in lightweight block cipher PIPO-64/128. The impossible differential characteristics search procedure is discussed and a full round related-key differential attack is presented. The paper is summarised with conclusion in Sect. 4.

2 MILP Based Differential Characteristic Search

To search the differential characteristics of a block cipher, the problem of optimizing the probability of differential characteristics is converted into the MILP problem. The objective function is the optimization of probabilities subject to the constraints based on linear inequalities. SPN and Feistel based block ciphers mainly consist of round key addition, substitution and permutation layers. The key addition layer does not contribute in the MILP model to search the differential characteristics. The input and output variables corresponding to the permutation layer are easily represented by linear inequalities. The substitution layer uses a non-linear S-box which cannot be easily represented by linear inequalities. SageMath is a popular tool that is used to obtain the linear inequalities using possible difference transitions in the DDT. In [31,32], Sasaki and Todo proposed the impossible transitions based approach to design a MILP problem to minimize the number of linear inequalities. This approach was later used by many researchers to design the MILP models of various 4-bit S-boxes [35]. The linear inequalities of permutation and substitution layers are used to model the MILP problem, that is solved by MILP solver GUROBI [15] or CPLEX [11].

In general, MILP based differential characteristics search is two stage process. Firstly, number of active S-boxes is minimized and then probability of differential characteristic is optimized using these active S-boxes. The outer and inner modules of MILP are designed corresponding to these stages. The outer module minimizes the number of active S-boxes while inner module optimizes the probabilities of differential characteristics.

2.1 Modeling Large S-box

An S-box is a non-linear component and its DDT is converted into linear inequalities to model the MILP problem. SageMath is used to generate the linear inequalities for DDT of the S-box. For m-bit S-box, the size of DDT is $2^m \times 2^m$ and it represents the number of occurrences of possible output differences corresponding to each input difference. SageMath uses the H-representation of convex hull to generate linear inequalities for the S-box. SageMath has practical-time limitation on the dimension of such convex hulls, so this method can be used to generate the linear inequalities for small S-boxes only. Therefore, this method cannot be used to model the outer module of MILP problem for S-boxes of size greater than 6 bits.

For large (8-bit) S-boxes, Abdelkhalek *et al.* [1] addressed this problem using the Espresso based tool Logic Friday [22] that reduces the inequalities by minimizing the product of sum of boolean functions. Boura and Coggia [8] proposed another method, inspired from QM algorithm, to reduce the number of linear inequalities of AES [26] and SKINNY [9] S-boxes. The proposed methods minimize the number of inequalities significantly in comparison to the existing approaches but at the cost of time and resources. The minimization process may take several days to get the reduced set of linear inequalities. The techniques

presented in [1] and [8], were used to minimize the number of active S-boxes for 8-bit S-box based ciphers.

To optimize the probability of differential characteristic, Abdelkhalek *et al.* used *pb*-DDT based approach by separating the DDT for each probability [1]. These DDTs are represented by 8-bit input difference and 8-bit output difference. The method was proposed due to limitation of logic friday to process the input with dimension more than 16. Although, this method is used to optimize the probability of differential characteristic, it may not be efficient due to the of use of *pb*-DDT instead of full DDT. Based on the full DDT of 4-bit S-box, Sun *et al.* [28] suggested the method of using extra variable for each unique probability. Using this method, linear inequalities will be generated in more than 16 input variable for 8-bit S-box. As Logic Friday is unable to handle more than 16 input variable, we use a Espresso based tool namely MILES[1] (Appendix B and C) that handles more than 16 variables to minimize the set of linear inequalities. Linear inequalities generated from MILES are used to design the outer and inner modules of MILP model which optimizes the probability of differential characteristic.

2.2 Linear Inequalities for Minimization of Active S-boxes

Constructing linear inequalities for S-box is the first step towards the MILP modeling of differential attack. To minimize the number of active S-boxes, the MILP model requires linear inequalities corresponding to all possible transitions in DDT. Some of the existing approaches e.g. H-representation of convex hull and QM algorithm are not time efficient for large ($n \geq$ 8-bit) S-boxes due to large dimension ($2n$). Espresso algorithm works efficiently for large S-boxes but provides a large set of minimized inequalities. To minimize the set of inequalities, Boura and Coggia [8] used prime implicants of QM algorithm to get an initial set of inequalities and proposed an algorithm to introduce a new set of inequalities. The combined set of linear inequalities is minimized by removing the impossible transitions [8,28]. The proposed method reduces the number of inequalities significantly but the time complexity to achieve this reduction is very high. Although, smaller MILP model with lesser inequalities does not guarantee a faster solution, yet MILP model having less number of inequalities is a preferred choice.

We present a new method[2] to minimize the number of linear inequalities for large S-box within the practical time limit. This method uses the output of QM algorithm partially and introduce a novel approach to add a better set of linear inequalities. The QM algorithm can be divided into three parts. In first part (QM_1), it constructs prime implicants from impossible transitions of the DDT. In second part (QM_2), prime implicants are reduced to get the essential prime implicants. These essential prime implicants are further reduced using the

[1] https://github.com/tarunyadav/MILES.
[2] https://github.com/tarunyadav/PIPO-MILP/tree/main/PIPO-MILP-Ineq-Reduction.

coverage approach in third part (QM_3). In our method, we use the output of QM_2 and introduce an inequality corresponding to each essential prime implicant $a = (a_0, a_1, \cdots, a_{n-1})$ (Eq. 1). Our method is applied in four phases as described in Algorithm 1. The set of initial inequalities (L) is constructed in phase 1 and a new set of linear inequalities is introduced using L in phase 2. For each impossible transition in DDT, we add the inequalities which remove that impossible transition. We introduce an additional inequality corresponding to all possible transitions in DDT. This inequality is constructed by adding all the inequalities in the set L. This inequality with new linear inequalities (L_{new}) are combined with initial set (L) to get a larger set of inequalities. In phase 3, we construct the MILP model to minimize the number of linear inequalities using the approach proposed in [28]. For each impossible transition, we add a constraint such that at least one inequality removing this transition remains in the minimized set. Using such constraints, we want to ensure that all impossible transitions are removed using the minimum number of linear inequalities. The objective of this MILP problem is minimization of the set of linear inequalities (L). In phase 4, we solve the MILP model using GUROBI solver to get the minimized set of linear inequalities (L_{min}).

$$\sum_{i=0}^{n-1}(1-a_i)x_i + a_i(1-x_i) \geq 1 \tag{1}$$

We compare the number of linear inequalities and time complexity of our algorithm with existing results for 8-bit S-boxes of AES, SKINNY-128 and PIPO-64/128 in Table 1. It is evident that our algorithm optimizes the trade-off between number of inequalities and time efficiency. For PIPO-64/128, we have solved the MILP model for active S-box minimization with 4476 inequalities constructed by MILES and 3276 inequalities constructed using Algorithm 1. The comparison of time to reach the lower bound is presented in Table 2. T_{OB} and T_{OS} represent the time to reach the optimal bound(OB) and time to conclude that the given optimal bound is the optimal solution respectively. It can be observed from Table 2 that the T_{OS} is always lesser for the larger set of inequalities(Model 1) which suggests that more constraints speeds up the process to eliminate impossible space. There is no such relation in T_{OB} that means smaller set of inequalities may not reach the optimal bound faster. The comparison concludes that lesser inequalities construct smaller model but may not always yield a faster solution.

2.3 Linear Inequalities for Optimization of Probability

Once active S-boxes are minimized, the next step is to optimize the probability corresponding to these active S-boxes. To optimize the probability, construction of linear inequalities corresponding to each possible probabilistic transition in DDT is required. To construct such linear inequalities, Abdelkhalek *et al.* [1] used the approach to construct separate DDT for each probability. These *pb*-DDTs are used to construct linear inequalities in the same manner as described in Sect. 2.2. Linear inequalities for each *pb*-DDT can be generated either using Espresso or

Algorithm 1. Linear Inequalities Minimization for 8-bit S-boxes

Input: 8-bit Sbox
Output: L_{min} = Set of minimized linear inequalities
1: $DDT \leftarrow$ Difference Distribution Table of S-box

Phase 1 - Initial Set of Linear Inequalities

2: Prime Implicants $\leftarrow QM_1$(Impossible Transitions of DDT)
3: Essential Prime Implicants $\leftarrow QM_2$(Prime Implicants)
4: $L \leftarrow$Linear inequalities corresponding to each essential prime implicant (Eqn 1)

Phase 2 - Adding New Inequalities

5: $L_{new} \leftarrow \phi$
6: **for all** Impossible Transitions $(X \rightarrow Y)$ in DDT **do**
7: $\quad S_{ineq} \leftarrow \{l_i, \forall l_i \in L$ and $l_i(X,Y) < 0\}$
8: $\quad L_{new} \leftarrow L_{new} \cup \{\sum S_i \ \forall S_i \in S_{ineq}\}$ ▷ Addition of coefficients
9: **end for**
10: $L \leftarrow L \cup L_{new} \cup \{\sum l_i \ \forall l_i \in L\}$

Phase 3 - MILP Modeling M (Objective and Constraints in Binary Variables)

11: $M.constraints \leftarrow \phi$
12: **for all** Impossible Transitions $(X \rightarrow Y)$ in DDT **do**
13: $\quad Z \leftarrow \{Z_i, \forall l_i \in L$ and $l_i(X,Y) < 0\}$ ▷ Z_i is a binary variable
14: $\quad M.constraints \leftarrow M.constraints \cup \{\sum Z \geq 1\ \}$
15: **end for**
16: $M.objective \leftarrow Min(\sum\{Z_i, \forall l_i \in L\})$

Phase 4 - Minimization using GUROBI Solver

17: $L_{min} \leftarrow M.optimize()$

Table 1. Comparison of time required to minimize linear inequalities of S-box

Algorithm	Output	S-box		
		AES	SKINNY	PIPO
QM	L_{min}	–	392	–
	Time	5 h	7707 s	5 h
Espresso	L_{min}	8389	377	4474
	Time	100 s	2 s	41 s
Algo 1 in [8]	L_{min}	7461	372	–
	Time	20d	120 m	–
Algo 3 in [8]	L_{min}	2882	302	–
	Time	Several days	120 m	–
Algorithm 1	L_{min}	5461	315	3276
	Time	5 h	900 s	5 h

Table 2. Comparison of time required (in seconds) to attain optimal bound/solution for PIPO-64/128 osing different sets of linear inequalities

Round	OB	Model 1			Model 2		
		Inequalities of S-box = 4474			Inequalities of S-box = 3276		
		Size (KB)	T_{OB}	T_{OS}	*Size (KB)*	T_{OB}	T_{OS}
1	1	4462	0.47	0.47	3902	0.81	0.81
2	2	8923	5.19	5.19	7802	2.00	9.32
3	4	13384	7.00	50.51	11704	4.00	745.23
4	6	17845	22.00	550.11	15602	28.00	8597.95
5	9	22305	208.00	–	19502	1546.00	–
6	11	26766	2290.00	–	23403	112.00	–

QM algorithm. QM based reduction depends on the characteristics of impossible transitions of the S-box and it needs to run for several days to provide a result. The algorithm proposed in [8] also suffers from similar drawback due to large number of impossible transitions in *pb*-DDTs. Due to the use of essential prime implicants in Algorithm 1, we get faster results than the existing approaches but still lack the time efficiency. There are some cases (Table 3) where the Algorithm 1 is not able to produce the result due to lesser number of possible transitions in DDT. The time complexity to produce the sets of minimized linear inequalities for each *pb*-DDT using MILES is less but each set contains the large number of inequalities. We compare the number of inequalities and execution time for *pb*-DDTs of AES, SKINNY and PIPO in Table 3.

The *pb*-DDT approach was proposed to overcome the limitation of Logic Friday since it becomes computationally infeasible to reduce the higher dimension inequalities of full DDT using Logic Friday. MILES uses the Espresso in its original form for reduction in higher dimension inequalities. Therefore, we can use the full DDT of S-box instead of *pb*-DDT. We use the approach proposed in [28] to construct the probability based possible transitions and introduce additional variable for each probability. We use MILES to construct the linear inequalities for these transitions and show that reduction using Espresso in higher dimension is faster than *pb*-DDT approach. Although, the use of full DDT may produce larger set of linear inequalities but it simplifies the MILP model as there is no need to choose different *pb*-DDT each time for an active S-box. We have already discussed (Table 2) that smaller set of linear inequalities doesn't guarantee the faster solution for optimal bound but may take more time to conclude the optimal solution. For PIPO-64/128, we use the approach of additional variables to utilize full DDT and apply the Espresso to construct a minimized set of inequalities. These inequalities will be used to optimize the probability of differential characteristics in the next section.

Table 3. Comparison of time required (in seconds) to get minimized set of linear inequalities to represent pb-DDT, p-TT and f-TT

Structure (S-box)	Probability	QM ([1]) (pb-DDT)	MILES (p-TT)		Algo 1 (p-TT)		MILES (f-TT)	
		L_{min}	L_{min}	Time	L_{min}	Time	L_{min}	Time
AES	2^{-7}	-	8312	84	5542	14963	8720	220
	2^{-6}	-	349	1	327	12478		
	Total	**-**	***8661***	***85***	***5869***	***27441***	***8720***	***220***
SKINNY	2^{-7}	206	208	1	187	8928	799	305
	2^{-6}	275	281	0.5	192	3903		
	$2^{-5.415037}$	33	34	0.3	-	-		
	2^{-5}	234	240	1.2	167	15903		
	$2^{-4.415037}$	42	47	0.2	-	-		
	2^{-4}	147	155	1.2	-	-		
	$2^{-3.678071}$	15	15	0.2	-	-		
	$2^{-3.415037}$	24	26	0.2	-	-		
	$2^{-3.192645}$	15	15	0.1	-	-		
	2^{-3}	62	69	0.3	-	-		
	$2^{-2.678071}$	16	16	0.1	-	-		
	$2^{-2.415037}$	17	17	0.1	-	-		
	2^{-2}	37	38	0.1	-	-		
	Total	***1123***	***1161***	***5.5***	**-**	**-**	***799***	***305***
PIPO	2^{-7}	-	3410	23	2464	14382	6035	2220
	2^{-6}	-	2211	24	1993	6446		
	$2^{-5.415037}$	-	519	5	479	7046		
	2^{-5}	-	355	7	294	10442		
	$2^{-4.678072}$	-	26	0.2	24	1566		
	$2^{-4.415037}$	-	20	0.1	20	1907		
	2^{-4}	-	93	0.5	57	10115		
	Total	**-**	***6634***	***59.8***	***5331***	***51904***	***6035***	***2220***

3 Application to Lightweight Block Cipher PIPO-64/128

Lightweight cryptography has become an important topic in cryptology [7] and NIST has also called for a competition to design the lightweight cryptographic primitives [25]. PIPO-64/128 is a lightweight block cipher which was recently proposed by Kim *et al.* at ICISC 2020 [17]. The design highlights are its security for side-channel protected and unprotected environments. Its diffusion layer is designed to optimize the efficiency in hardware as well as software applications. Its diffusion layer can be implemented in software using the cyclic shift rotations.

For hardware applications, its diffusion layer can be visualised as bit permutation on 64 bits and can be implemented using wiring only. The 8-bit S-box is specifically designed for PIPO-64/128 so that it can be represented using the minimum number of non-linear equations. This also ensures the protection of the design against side channel attacks.

3.1 Specification of PIPO-64/128

PIPO-64/128 is a 64-bit lightweight block cipher with 128 and 256 bits key sizes [17]. It consists of 13/17 rounds for 128/256 bits key variants respectively. It is based on substitution permutation network (SPN) structure. The lightweight 8-bit S-box, having differential branch number 3, is specifically designed to use in the confusion layer of PIPO-64/128. For each 8-bit word, diffusion layer uses a cyclic rotation with different shift values for each word. The round function of PIPO-64/128 is explained by dividing it into an 8×8 matrix. It applies the diffusion layer row-wise and 8-bit S-box is applied column-wise. For each variant, a simple key selection algorithm is used. For 128-bit key $K = (K_1 \parallel K_0)$, the rounds keys are selected as $RK_i = K_{i(mod2)}, 0 \leq i \leq 13$. For 256-bit key $K = (K_3 \parallel K_2 \parallel K_1 \parallel K_0)$, the rounds keys are selected as $RK_i = K_{i(mod4)}$, $0 \leq i \leq 17$.

Algorithm 2. Encryption Algorithm of PIPO-64/128

Input: P and $RK_i, 0 \leq i \leq 13$
Output: $C = (c_{63}, c_{62}, \cdots, c_0)$
1: $U_0 \leftarrow P \oplus RK_0$
2: $U_0 = (u_{63}, u_{62}, \cdots, u_0)$
3: **for** i=1 to 13 **do**
4: **for** j=0 to 7 **do**
5: $(v_{56+j} \parallel v_{48+j} \parallel v_{40+j} \parallel v_{32+j} \parallel v_{24+j} \parallel v_{16+j} \parallel v_{8+j} \parallel v_j)$
6: $\leftarrow S_8(u_{56+i} \parallel u_{48+i} \parallel u_{40+j} \parallel u_{32+j} \parallel u_{24+j} \parallel u_{16+j} \parallel u_{8+j} \parallel u_j)$
7: **end for**
8: **end for**
9: $V_i = (v_{63}, v_{62}, \cdots, v_0)$
10: $U_i \leftarrow B_P(V_i) \oplus RK_i \oplus i$
11: $U_i = (u_{63}, u_{62}, \cdots, u_0)$

For MILP modeling, we describe the encryption algorithm of PIPO-64/128 in a different way (Algorithm 2). Round function is described using substitution layer, permutation layer and add round key operations. Substitution layer applies 8-bit S-box (S) (Table 4) on 8 bits extracted from eight different positions of input and output bits from S-box are sent back to the same positions.

Permutation layer uses a 64-bit permutation (B_P) (Table 5) on the output from S-box layer. The round keys (RK_i) and constants ($i = round\,number$) are simply XOR-ed with the output of diffusion layer.

Table 4. 8-bit S-box of PIPO-64/128

S_8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	5E	F9	FC	00	3F	85	BA	5B	18	37	B2	C6	71	C3	74	9D
1	A7	94	0D	E1	CA	68	53	2E	49	62	EB	97	A4	0E	2D	D0
2	16	25	AC	48	63	D1	EA	8F	F7	40	45	B1	9E	34	1B	F2
3	B9	86	03	7F	D8	7A	DD	3C	E0	CB	52	26	15	AF	8C	69
4	C2	75	70	1C	33	99	B6	C7	04	3B	BE	5A	FD	5F	F8	81
5	93	A0	29	4D	66	D4	EF	0A	E5	CE	57	A3	90	2A	09	6C
6	22	11	88	E4	CF	6D	56	AB	7B	DC	D9	BD	82	38	07	7E
7	B5	9A	1F	F3	44	F6	41	30	4C	67	EE	12	21	8B	A8	D5
8	55	6E	E7	0B	28	92	A1	CC	2B	08	91	ED	D6	64	4F	A2
9	BC	83	06	FA	5D	FF	58	39	72	C5	C0	B4	9B	31	1E	77
A	01	3E	BB	DF	78	DA	7D	84	50	6B	E2	8E	AD	17	24	C9
B	AE	8D	14	E8	D3	61	4A	27	47	F0	F5	19	36	9C	B3	42
C	1D	32	B7	43	F4	46	F1	98	EC	D7	4E	AA	89	23	10	65
D	8A	A9	20	54	6F	CD	E6	13	DB	7C	79	05	3A	80	BF	DE
E	E9	D2	4B	2F	0C	A6	95	60	0F	2C	A5	51	6A	C8	E3	96
F	B0	9F	1A	76	C1	73	C4	35	FE	59	5C	B8	87	3D	02	FB

Table 5. Bit permutation in PIPO-64/128

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$B_P(i)$	0	1	2	3	4	5	6	7	15	8	9	10	11	12	13	14
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$B_P(i)$	20	21	22	23	16	17	18	19	27	28	29	30	31	24	25	26
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$B_P(i)$	38	39	32	33	34	35	36	37	45	46	47	40	41	42	43	44
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$B_P(i)$	49	50	51	52	53	54	55	48	58	59	60	61	62	63	56	57

Table 6. Difference distribution table of PIPO-64/128

$\triangle_j \rightarrow$																
$\triangle_i \downarrow$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	$\cdots$	FF
0	256	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$	0
2	0	0	0	0	0	16	0	0	0	0	0	0	0	0	$\cdots$	0
3	0	0	0	0	0	16	0	0	0	0	0	0	0	0	$\cdots$	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$	0
6	0	0	0	0	0	16	0	0	0	0	0	0	0	0	$\cdots$	0
7	0	0	0	0	0	16	0	0	0	0	0	0	0	0	$\cdots$	0
$\cdots$															$\cdots$	
8F	0	0	0	0	0	8	0	0	0	8	0	0	0	0	$\cdots$	0
9F	0	0	16	0	0	0	0	4	2	0	0	0	0	0	$\cdots$	0
AF	0	2	16	0	0	0	0	0	0	0	0	2	0	0	$\cdots$	0
BF	0	2	0	0	0	0	0	4	0	0	0	2	0	0	$\cdots$	0
CF	0	2	0	0	0	0	0	0	2	0	0	0	0	4	$\cdots$	2
DF	0	2	16	0	0	0	0	4	0	0	4	0	0	4	$\cdots$	2
EF	0	0	16	0	0	0	0	0	2	0	0	0	2	0	$\cdots$	0
FF	0	0	0	0	0	0	0	4	0	0	2	0	0	0	$\cdots$	2

3.2 MILP Modeling for PIPO-64/128

The model for valid differential propagations of PIPO-64/128 is constructed bit-wise. In each round, subkey addition, S-box, and bit permutation operations are used. Block size in PIPO-64/128 is 64-bit and it consists of 13 rounds. For 64-bit plaintext difference, binary variables $u_{63}, u_{62}, \cdots u_0$ represent active or inactive bits for first round. The variables to represent active or inactive bits in the difference after first round are updated to $u_{127}, u_{126}, \cdots u_{64}$ and so on. The variables $u_{832}, u_{831}, \cdots u_{768}$ represent the active or inactive bits in the ciphertext difference after 13 rounds. In first round, the variables representing the bits of input and output differences to S-box layer are represented as follows:

$$\begin{bmatrix} u_7 & u_6 & u_5 & u_4 & u_3 & u_2 & u_1 & u_0 \\ u_{15} & u_{14} & u_{13} & u_{12} & u_{11} & u_{10} & u_9 & u_8 \\ u_{23} & u_{22} & u_{21} & u_{20} & u_{19} & u_{18} & u_{17} & u_{16} \\ u_{31} & u_{30} & u_{29} & u_{28} & u_{27} & u_{26} & u_{25} & u_{24} \\ u_{39} & u_{38} & u_{37} & u_{36} & u_{35} & u_{34} & u_{33} & u_{32} \\ u_{47} & u_{46} & u_{45} & u_{44} & u_{43} & u_{42} & u_{41} & u_{40} \\ u_{55} & u_{54} & u_{53} & u_{52} & u_{51} & u_{50} & u_{49} & u_{48} \\ u_{63} & u_{62} & u_{61} & u_{60} & u_{59} & u_{58} & u_{57} & u_{56} \end{bmatrix} \rightarrow \begin{bmatrix} u_{71} & u_{70} & u_{69} & u_{68} & u_{67} & u_{66} & u_{65} & u_{64} \\ u_{78} & u_{77} & u_{76} & u_{75} & u_{74} & u_{73} & u_{72} & u_{79} \\ u_{83} & u_{82} & u_{81} & u_{80} & u_{87} & u_{86} & u_{85} & u_{84} \\ u_{90} & u_{89} & u_{88} & u_{95} & u_{94} & u_{93} & u_{92} & u_{91} \\ u_{101} & u_{100} & u_{99} & u_{98} & u_{97} & u_{96} & u_{103} & u_{102} \\ u_{108} & u_{107} & u_{106} & u_{105} & u_{104} & u_{111} & u_{110} & u_{109} \\ u_{112} & u_{119} & u_{118} & u_{117} & u_{116} & u_{115} & u_{114} & u_{113} \\ u_{121} & u_{120} & u_{127} & u_{126} & u_{125} & u_{124} & u_{123} & u_{122} \end{bmatrix}$$

The permutation layer is applied on the output from S-box layer and output of the permutation layer which acts as an input to the second round is represented as follows:

$$\begin{bmatrix} u_{71} & u_{70} & u_{69} & u_{68} & u_{67} & u_{66} & u_{65} & u_{64} \\ u_{79} & u_{78} & u_{77} & u_{76} & u_{75} & u_{74} & u_{73} & u_{72} \\ u_{87} & u_{86} & u_{85} & u_{84} & u_{83} & u_{82} & u_{81} & u_{80} \\ u_{95} & u_{94} & u_{93} & u_{92} & u_{91} & u_{90} & u_{89} & u_{88} \\ u_{103} & u_{102} & u_{101} & u_{100} & u_{99} & u_{98} & u_{97} & u_{96} \\ u_{111} & u_{110} & u_{109} & u_{108} & u_{107} & u_{106} & u_{105} & u_{104} \\ u_{119} & u_{118} & u_{117} & u_{116} & u_{115} & u_{114} & u_{113} & u_{112} \\ u_{127} & u_{126} & u_{125} & u_{124} & u_{123} & u_{122} & u_{121} & u_{120} \end{bmatrix}$$

We describe all possible propagation patterns for S-box with a system of linear inequalities.

$$e.g.(u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0 \rightarrow u_{71}, u_{70}, u_{69}, u_{68}, u_{67}, u_{66}, u_{65}, u_{64})$$

The variables corresponding to bits having the difference takes '1' and it takes '0' otherwise. A constraint $u_0 + u_1 + \cdots + u_{63} \geq 1$ is added to ensure that plaintext difference has at least one active bit.

Modeling 8-bit S-box. To model the 8-bit S-box of PIPO-64/128, we generate the DDT (Table 6) for each possible input and output difference (Δ_i, Δ_j) using MILES. The entries (i, j) in the Table 6 corresponds to the number of occurrences for output differences Δ_j when the input differences were set as Δ_i. We get a 256×256 DDT for an 8-bit S-box. The non-zero values in the DDT corresponds to a possible difference propagation and zero values indicates an impossible propagation.

Linear Inequalities for Outer Module of MILP Model. The DDT generated in previous step is used in MILES to derive the truth table ($\star$-TT). The $\star$-TT of PIPO-64/128 contains 20621 entries which are further minimized by our tool. MILES minimizes the $\star$-TT to $\star$-TT$_{min}$ with 4474 entries. We convert each entry of $\star$-TT$_{min}$ into a linear inequality. We represent each entry of $\star$-TT$_{min}$ using 16 binary variables $(x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7)$, where first eight variables $(x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ represent the input difference and remaining variables $(y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7)$ represent the output difference. These linear inequalities are used as constraints in the outer module and minimization of number of active S-boxes is used as objective function.

Linear Inequalities for Inner Module of MILP Model. Differential probability of S-box was used to design MILP model by Sun *et al.* in [27] and this technique was also used by Zhu *et al.* to present the MILP based differential attack on round-reduced GIFT in [35]. We optimize the probability of differential characteristics in the inner module of MILP model. For this purpose, we need the linear inequalities for all non-zero entries in the DDT which corresponds to

the possible difference propagation and their probabilities. In the DDT of PIPO-64/128 S-box, there are seven different values for the probability of possible difference propagations i.e. 2^{-0}, $2^{-4.00}$, $2^{-4.41}$, $2^{-4.67}$, $2^{-5.00}$, $2^{-5.41}$, $2^{-6.00}$, $2^{-7.00}$ (Table 7). This requires seven extra binary variables to represent the probability of each possible propagation. MILES uses DDT to generate truth table (f-TT) with 20621 entries. Each entry of the f-TT is represented by 23 binary variables where 16 input variables $(x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7)$ represents the input and output differences. The remaining seven input variables $(p_0, p_1, p_2, p_3, p_4, p_5, p_6)$ represent the probabilities of corresponding difference propagations. MILES minimizes the f-TT to f-TT$_{min}$ which results in 6035 entries in f-TT$_{min}$. Each entry of f-TT$_{min}$ is converted into the linear inequality using Eq. 1. This set of linear inequalities is used to optimize the probability of differential characteristics in the block cipher PIPO-64/128.

Table 7. Binary variables to encode the probabilities in DDT of PIPO-64/128

$Pr[(x_0, x_1, \cdots, x_7) \rightarrow (x_8, x_9, \cdots, x_{15})]$	$(p_0, p_1, \cdots, p_6)$
$1 = 2^{-0}$	(0,0,0,0,0,0,0)
$2/256 = 2^{-7.00}(Pr_6)$	(0,0,0,0,0,0,1)
$4/256 = 2^{-6.00}(Pr_5)$	(0,0,0,0,0,1,0)
$6/256 = 2^{-5.41}(Pr_4)$	(0,0,0,0,1,0,0)
$8/256 = 2^{-5.00}(Pr_3)$	(0,0,0,1,0,0,0)
$10/256 = 2^{-4.67}(Pr_2)$	(0,0,1,0,0,0,0)
$12/256 = 2^{-4.41}(Pr_1)$	(0,1,0,0,0,0,0)
$16/256 = 2^{-4.00}(Pr_0)$	(1,0,0,0,0,0,0)

3.3 Differential Cryptanalysis of PIPO-64/128

We solve the MILP model using Gurobi solver [15] to optimize the probability of differential characteristics for PIPO-64/128. In the outer-MILP module, the objective function is to minimize the number of active S-boxes in the differential characteristics. We get 13 active S-boxes for 7 rounds differential characteristics in PIPO-64/128. The objective function for the inner-MILP module is to maximize the probability of differential characteristics using the positions of active S-boxes obtained in the outer module. The objective function is defined as minimization of Eq. 2 over active S-boxes (AS).

$$\sum_{\forall AS} \sum_{i=0}^{6} -\log_2(Pr_i) \times (p_0 + p_1 + p_2 + p_3 + p_4 + p_5 + p_6) \tag{2}$$

We constructed[3] many differential characteristics for PIPO-64/128 reduced to 6/7 rounds. There does not exist any 6-round differential characteristic with the probability better than $2^{-54.4}$ and best differential characteristics for 7-round PIPO-64/128 exists with the probability of 2^{-65}. We constructed the 7-round differential characteristics for PIPO-64/128 using the inequalities generated with MILES which is shown in Table 8.

Table 8. 7-round differential characteristics for PIPO-64/128

Round (r)	Input difference (Δ_r)	Probability
0	0x0101000101000001	1
1	0x0000000000008000	2^{-4}
2	0x0000000000080080	2^{-4}
3	0x2011112000800080	2^{-11}
4	0x404100408101c080	2^{-19}
5	0x0000101000100000	2^{-16}
6	0x0000000080000000	2^{-7}
7	0x0001000004084000	2^{-4}

3.4 Impossible Differential Cryptanalysis of PIPO-64/128

Impossible differential attack is opposite to differential attack. The basic idea is to use zero probability differential characteristics in place of a high probability characteristic to filter out the wrong keys [3]. For this purpose, the zero probability characteristics are constructed by proving a contradiction between the two differential characteristics of probability one each. This approach is known as miss-in-the-middle technique to search an impossible differential characteristic. Nowadays, the MILP based technique is used to search these zero probability differential characteristics. The MILP model to search the high probability differential characteristics with some added constraint is used to search the impossible differential characteristic.

To search the impossible differential, we iterate all (Δ_i, Δ_o) pairs with one active bit in the input and output. For this purpose, additional constraints to fix the input and output differences are added in the MILP model. The gurobi solver is used to solve the outer module of MILP model as discussed in Sect. 3.2. The input and output differences corresponding to infeasible solution are considered as impossible differential characteristic. Using this method[4], we obtain the

[3] https://github.com/tarunyadav/PIPO-MILP.
[4] https://github.com/tarunyadav/PIPO-MILP/tree/main/PIPO-MILP-Impossible-Differential.

following 4-round impossible differential characteristics (Δ_0, Δ_4). However, our bound for impossible differential attack is similar to that of the designers claim.

$$0000000000000000000000000000000000000000000000000000000000000001 \rightarrow$$
$$0000000000000000000000000000000000000000000000000000000000010000$$

3.5 Related-Key Differential Distinguisher for PIPO-64/128

Resistance to related-key attacks was not considered by the designers of PIPO-64/128 and any security claim in the related-key setting is not provided. In differential attack, the adversary is allowed to choose a difference in the plaintexts and observe the differences in ciphertexts. In related-key differential attack, the adversary is allowed to choose a relation (difference) in the key together a relation (difference) in the plaintexts [2,21]. The adversary is allowed to get the encryption of first plaintext using the secret key and a key related to this key is used to encrypt the another plaintext. We model an MILP problem to search the related-key differential characteristic in PIPO-64/128.

3.5.1 MILP Model for Related-Key Differential Characteristic. The secret key K is divided into the two 64-bit keys K_0 and K_1 which are used as round subkeys in PIPO-64/128. We model the similar MILP problem to search the related-key differential characteristic as described in Sect. 3.2. Additionally, we need to model the key addition layer and solve the MILP model in order to get the optimal related-key characteristics in PIPO-64/128 [27].

Modeling Key Addition Layer. We need to introduce the additional constraints in the MILP model corresponding to the round keys. The 128 new variables are introduced corresponding to the 128-bit secret key. The 64 key variables are added in one round and the other 64 key variables are added in the subsequent round. To add the constraints for key addition layer, for each bit of input x_i and key k, we follow the conditions on bit variables to exclude the impossible patterns (Eq. 3). Here, x_i and k refer to the input bit and corresponding key bit. The bit variable x_{i+1} is an output of the XOR operation *i.e.* $x_{i+1} = k \oplus x_i$.

$$\begin{aligned} x_i + k - x_{i+1} &\geq 0 \\ x_i - k + x_{i+1} &\geq 0 \\ -x_i + k + x_{i+1} &\geq 0 \\ x_i + k + x_{i+1} &\leq 2 \end{aligned} \tag{3}$$

3.5.2 Full-Round Related-Key Differential Distinguisher. We solve[5] the MILP model to search the related-key differential characteristics using gurobi solver. We get a 2-round iterative related-key characteristic with the probability

[5] https://github.com/tarunyadav/PIPO-MILP/tree/main/PIPO-MILP-Related-Key-Differential.

of 2^{-4}. The optimal related-key differential characteristic for full round PIPO-64/128 is obtained with a probability of 2^{-24} using 2-round iterative characteristic (Table 9). We also get full-round characteristics with probability of 2^{-28} under zero difference in the plaintext as well as in the ciphertext (Table 10).

3.5.3 Collisions in PIPO-64/128. The zero difference related-key characteristics will lead to a collision in the hash function designed using PIPO-64/128. We searched for the existence of input and output pairs under different keys following zero difference characteristic (collision). We encrypt the 2^{28} random samples under related keys and one such pair is expected in each experiment. Therefore, we can construct as many samples providing us the collision in the input and output under the different keys. We have verified these plaintext and ciphertext samples by using the designers program. One such collision in PIPO-64/128 is presented in the Table 10. We have also provided other samples showing a collision in the Appendix A.

Table 9. 13-round (related-key) differential characteristic for PIPO-64/128 with probability 2^{-24}

Round (r)	Difference (Δ_r) ΔK= 0x00200000200000000040000801001000	Probability
0	0x0040000801001000	1
1	0x0020000020000000	1
2	0x0000000000000000	2^{-4}
3	0x0020000020000000	1
4	0x0000000000000000	2^{-4}
5	0x0020000020000000	1
6	0x0000000000000000	2^{-4}
7	0x0020000020000000	1
8	0x0000000000000000	2^{-4}
9	0x0020000020000000	1
10	0x0000000000000000	2^{-4}
11	0x0020000020000000	1
12	0x0000000000000000	2^{-4}
13	0x0020000020000000	1

Table 10. Zero difference characteristics with an example of collision

Round (r)	$E_K(P_r)$, $K = (K_1 \| K_0)$ K_1= 0x6DC416DD779428D2 K_0= 0x7E1D20AD2E152297	$E_{K'}(P_r')$, $K' = (K_1' \| K_0')$ $K_1' = K_1 \oplus$ 0x0040000801001000 $K_0' = K_0 \oplus$ 0x0020000020000000	Difference ($\Delta_r = P_r \oplus P_r'$)	Probability
0	0xFFEAF697D7FCE742	0xFFEAF697D7FCE742	0x0000000000000000	1
1	0xD76EFD65756940C0	0xD76EFD65756940C0	0x0000000000000000	2^{-4}
2	0x4FA59C5858EDC4FF	0x4F859C5878EDC4FF	0x0020000020000000	1
3	0x8F6ACEC7A220C121	0x8F6ACEC7A220C121	0x0000000000000000	2^{-4}
4	0x406AD151D57A997B	0x404AD151F57A997B	0x0020000020000000	1
5	0xC5F53C44C408AC2D	0xC5F53C44C408AC2D	0x0000000000000000	2^{-4}
6	0xCFD8867C58BFCFD9	0xCFF8867C78BFCFD9	0x0020000020000000	1
7	0xC99B445F8E203697	0xC99B445F8E203697	0x0000000000000000	2^{-4}
8	0xD12CCC87E5585504	0xD10CCC87C5585504	0x0020000020000000	1
9	0x01D75CDC373A6F41	0x01D75CDC373A6F41	0x0000000000000000	2^{-4}
10	0x41C1CE1756D7C045	0x41E1CE1776D7C045	0x0020000020000000	1
11	0x388794675E6B5EDE	0x388794675E6B5EDE	0x0000000000000000	2^{-4}
12	0x1FDB4194BF26AC3B	0x1FFB41949F26AC3B	0x0020000020000000	1
13	0xCDE57DF09ECF4F7D	0xCDE57DF09ECF4F7D	0x0000000000000000	2^{-4}

3.6 Related-Key Differential Attack on Full-round PIPO-64/128

We use the related-key differential characteristic described in the Table 10 to present a full-round differential attack on PIPO-64/128. We used 11-round differential characteristics ($\Delta_1 \rightarrow \Delta_{12}$) with the probability of $2^{-20}$and added one round at the beginning as well as at the end of the characteristic (Table 11). Using the 11-round differential characteristic, we can launch a key recovery attack on the 13-round PIPO-64/128. The 11-round characteristic is chosen in particular to maximize the number of recovered key bits. In each round, 64-bit round key is required and it is extracted directly from the 128-bit key $K = (K_1, K_0)$. The key K_0 is used for whitening and for even numbered rounds while the odd numbered rounds use the key K_1. We need to guess the round keys which correspond to the actives S-boxes.

Table 11. Related-key differential attack on 13-round PIPO-64/128

$\Delta K_1 \rightarrow$	0000 0000 0100 0000 0000 0000 0000 1000 0000 0001 0000 0000 0001 0000 0000 0000
$\Delta K_0 \rightarrow$	0000 0000 0010 0000 0000 0000 0000 0000 0010 0000 0000 0000 0000 0000 0000 0000
Δ_0	00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000
$\oplus \Delta K_0$	00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000
S-Box	0000 0000 0010 0000 0000 0000 0010 0000 0010 0000 0000 0000 0010 0000 0000 0000
Permutation	0000 0000 0100 0000 0000 0000 0000 1000 0000 0001 0000 0000 0001 0000 0000 0000
Δ_1	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
.	
.	
.	
Δ_{12}	0000 0000 0010 0000 0000 0000 0000 0000 0010 0000 0000 0000 0000 0000 0000 0000
S-Box	00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000 00?0 0000
Permutation	?000 0000 0?00 0000 0000 0?00 0000 ?000 0000 000? 0000 00?0 000? 0000 00?0 0000
Δ_{13}	?000 0000 0?00 0000 0000 0?00 0000 ?000 0000 000? 0000 00?0 000? 0000 00?0 0000

$\Delta_1 = Permutation \oplus K_1; \Delta_{13} = Permutation \oplus K_1$

3.6.1 Data Collection

We can build $2^n (n \leq 56)$ structures corresponding to the fixed bits in the input difference (Δ_0). The objective is to minimize the value of n such that sufficient number of right pairs are left for key guessing phase. Each structure traverses the 8 undetermined (?) bits in Δ_0 (Table 11). Thus, each structure generates $2^{8*2-1} (= 2^{15})$ pairs[6] satisfying the differential. Therefore, the total number of pairs generated by the 2^n structures are 2^{n+15}. In Table 11, such a pair will meet the second round differential with an average probability of 2^{-8}. The probability of obeying the differential after 12^{th} round for the pair encrypted with the right key is 2^{-20}. Therefore, the number of pairs satisfying the differential after 12^{th} round for a right key guess will be $2^{n+15} \times 2^{-8} \times 2^{-20} (= 2^{n-13})$. Hence, we choose $n = 17$ so that we could get at least $2^4 (= 16)$ right pairs under the correct key guessing.

3.6.2 Key Recovery

In this phase, we guess the key bits corresponding to the undetermined bits(?) in Δ_0 and Δ_{13} and nonzero fix difference. This guess includes $K_0^5, K_0^{13}, K_0^{21}, K_0^{29}, K_0^{37}, K_0^{45}, K_0^{53}, K_0^{61}, K_1^{12}, K_1^{24}, K_1^{35}, K_1^{54}$ in 1^{st} round and $K_1^5, K_1^{12}, K_1^{17}, K_1^{24}, K_1^{35}, K_1^{42}, K_1^{54}, K_1^{63}$ in 13^{th} round. Since $K_1^{12}, K_1^{24}, K_1^{35}, K_1^{54}$ are involved in 1^{st} and 13^{th} round, total 16 unique key bits are involved in the key recovery phase. Hence, we construct 2^{16} counters corresponding to the possible values of 16 bits of the key.

With n = 17, we repeat the key guessing procedure for each of the $2^{17+15} (= 2^{32})$ pairs. We experimented with 2^{32} pairs and find that there are at least 2^4 pairs remaining after filtered by zero difference in Δ_{13}. Therefore, the expected counter value for a wrong key guess will be $2^{4-8-8} (= 2^{-12})$ after filtered by the undetermined bits in Δ_0 and Δ_{13}. As discussed in Sect. 3.6.1, there are at least 16 right pairs remaining after 12^{th} round. These right pairs will be used for key guessing and a key with the highest counter value will be the correct key.

3.6.3 Complexity

There are 2^n structures and 2^8 pairs(fixing the undetermined bits in Δ_0) can be generated for each structure. As discussed in Sect. 3.6.2, we need 2^{32} pairs to get 2^4 right pairs. Therefore, we choose n(=24) structure and the data complexity of the 13-round related-key differential attack on PIPO-64/128 becomes $2^{24+8} (= 2^{32})$. We need to store the counters corresponding to 16 bits of the key, so the memory complexity of the attack becomes 2^{16}. In the first round, for each of the 2^4 pairs, we need to guess the 12 bits of the key corresponding to the active S-box. Therefore, time complexity of the first round becomes $2^{4+12} (= 2^{16})$. Similarly time complexity of the 13^{th} round is $2^{4+4} (= 2^8)$ because four bits of the key are already guessed in the first round. Hence, the time complexity of the whole attack is bounded by the 2^{32} chosen plaintexts.

[6] In this calculation, we consider a pair (a, b) same as (b, a).

4 Conclusion

In this paper, we have presented the approaches to construct the linear inequalities corresponding to the DDT of 8-bit S-boxes. These inequalities are used to minimize the number of active S-boxes in PIPO-64/128. The experimental results indicate that there is no significant difference in the time complexity to solve the MILP models with a smaller set of linear inequalities. Therefore, we have used full DDT to construct a simplified MILP model for probability optimization instead of using the existing *pb*-DDT approach. The linear inequalities corresponding to the full DDT of PIPO-64/128 are constructed using the MILES tool. These linear inequalities are used to model the MILP problem for searching the differential, impossible differential and related-key differential characteristics. We have presented the full-round related-key differential distinguisher and a key recovery attack on full-round PIPO-64/128 with 2^{32} data complexity. We have also presented several collisions in the plaintext and ciphertext using different keys.

Appendix

A $\boldsymbol{C = E(P, K) = E(P, K') \text{ where } K' = K \oplus \Delta K}$
$\boldsymbol{K = 0x6DC416DD779428D27E1D20AD2E152297}$
$\boldsymbol{\Delta K = 0x00400008010010000020000020000000}$

No.	Plaintext (P)	Ciphertext (C)
1	0xFFEAF697D7FCE742	0xCDE57DF09ECF4F7D
2	0xFCFFE1E57B3EE1B0	0x964DFE673B256413
3	0xFE9DAF4B7CDF3C62	0x5A204F91F5B3BEE2
4	0xBFE622F4EDF3FF2A	0x2C41558C8D728AD0
5	0xE7FFA8E4E8F95AF5	0xEB10BDFF059CF6A0
6	0xBDFDE7BAFFF6E73E	0x009AEE178347B174
7	0x7FFB2EFE657B19E7	0xD387F51CC4D0755A
8	0x2FF9393C75FB73F1	0x46B43D51ABE5146D
9	0x6EFE60A8EFFF5F2F	0x4F687CEC564569ED
10	0xF8EFFEB4EFFC9A70	0x923B7FDBAE0812CC
11	0xFD976646A1A3B40C	0xC433269EE6751443
12	0x6FF431B77B748CB5	0x5041B64C120B2673
13	0xE3EBED217F6FEB3F	0x56072F13AA0DB152
14	0xE35BF593EB9D32F0	0x1046EFDED93A860F
15	0xFFF76DCC8F77FA1B	0x73D7C7FFE4A78EF6
16	0x77FF282B3F7F8121	0xAD7D75F547410892
17	0x3E6FAB372BFB5F23	0x17C097CDE69D86BA
18	0xEFFABDB4F6F7032E	0x98731593F9EFC0D7
19	0x75926BBA4F77726F	0xDF4974E78B9FEC13
20	0xEBE465797D6BAD63	0x7432FC827038315B

B MILES: MInimized Linear inEqualities for Large S-Boxes

We present expresso based tool MILES to generate the linear inequalities for larges S-boxes and this tool is based on the Espresso algorithm [12]. The S-box is given as an input to the tool and it outputs a minimized set of linear inequalities that is required to model the MILP problem. MILES is the first tool that uses the full DDT of 8-bit S-box to generate the linear inequalities. In MILES, there are four processes which are applied sequentially to generate the minimized linear inequalities. These process are described as follows:

1. **DDT generation.** In this process, MILES takes m-bit S-box (m≥3) as input and generates a DDT of the S-box. The DDT ($2^m \times 2^m$) is 2-Dimensional array where row indices (y-axis) define input difference while column indices (x-axis) define the output difference. We define a function $f_{i,j}$ to represent the DDT of S-box which provides the number of occurrences of output difference Δ_j corresponding to input difference Δ_i (Eq. 1).

$$f_{i,j} = Frequency_{\Delta_i \to \Delta_j} \text{where } 0 \leq i, j \leq m \tag{4}$$

   This DDT is used as an input in the next process.
2. **DDT to truth table conversion.** In this process, the input DDT is converted into a truth table. This truth table specifies the input and output points of the DDT as input variables. To simplify it, we specify only non-zero entries of the DDT and corresponding output variable as 1. MILES can generate three kinds of truth tables ($\star$-TT,p-TT,f-TT) from the DDT. The $\star$-TT table corresponds to the non-zero entries in the DDT and p-TT corresponds to the non-zero entries in DDT for a specific probability (p). The f-TT table corresponds to the non-zero entries with extra input variable for each probability.
3. **Truth table minimization.** MILES interfaces with Espresso to perform minimization of the truth table. The output of minimization is TT_{min} which is used to generate the minimized linear inequalities. The TT_{min} is similar to the truth table and it contains an additional symbol ('−'). The output variable in TT_{min} is independent of input variable corresponding to this additional symbol. The minimization process can be performed with various modes available in Espresso algorithm. These options are chosen in MILES as minimization strategy. These strategies are problem specific and a particular strategy may not provide best solution for all problems. The minimized truth tables corresponding to $\star$-TT, p-TT, and f-TT are represented as $\star$-TT_{min}, p-TT_{min}, and f-TT_{min} respectively.
4. **Linear inequalities generation.** After minimization process, MILES generate the linear inequalities. Each linear inequality corresponds to one entry in TT_{min}. If a value in the entry is 0 then it is expressed as variable x and if it is 1 then it is expressed as $1 - x$. The value '−' in the TT_{min} does not contribute in the inequality generation process.

C Example: Linear Inequalities Generation using MILES

We describe the process to generate the linear inequalities for a 3-bit S-box (Table 12). The DDT (Table 13), f-TT (Table 14), and f-TT$_{min}$ (Table 15) are generated using MILES. The set of minimized linear inequalities for this S-box is given in Table 16.

Table 12. 3-bit S-box

x	0	1	2	3	4	5	6	7
S(x)	3	6	5	7	0	2	4	1

Table 13. DDT of S-box

	0	1	2	3	4	5	6	7
0	8	0	0	0	0	0	0	0
1	0	0	4	0	0	4	0	0
2	0	2	0	2	2	0	2	0
3	0	2	0	2	2	0	2	0
4	0	2	0	2	2	0	2	0
5	0	2	0	2	2	0	2	0
6	0	0	0	0	0	4	0	4
7	0	0	4	0	0	0	0	4

Table 14. f-TT of DDT

x_1	x_2	x_3	y_1	y_2	y_3	p_1	p_2	f
0	0	0	0	0	0	0	0	1
0	0	1	0	1	0	1	0	1
0	0	1	1	0	1	1	0	1
0	1	0	0	0	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	0	1	0	0	0	1	1
0	1	0	1	1	0	0	1	1
0	1	1	0	0	1	0	1	1
0	1	1	0	1	1	0	1	1
0	1	1	1	0	0	0	1	1
0	1	1	1	1	0	0	1	1
1	0	0	0	0	1	0	1	1
1	0	0	0	1	1	0	1	1
1	0	0	1	0	0	0	1	1
1	0	0	1	1	0	0	1	1
1	0	1	0	0	1	0	1	1
1	0	1	0	1	1	0	1	1
1	0	1	1	0	0	0	1	1
1	0	1	1	1	0	0	1	1
1	1	0	1	0	1	1	0	1
1	1	0	1	1	1	1	0	1
1	1	1	0	1	0	1	0	1
1	1	1	1	1	1	1	0	1

Table 15. f-TT$_{min}$ for f-TT

x_1	x_2	x_3	y_1	y_2	y_3	p_1	p_2	f
0	0	1	0	1	0	1	0	1
0	0	1	1	0	1	1	0	1
1	1	1	0	1	0	1	0	1
1	1	0	1	-	1	1	0	1
1	1	-	1	1	1	1	0	1
0	0	0	0	0	0	0	0	1
1	0	-	1	-	0	0	1	1
0	1	-	1	-	0	0	1	1
1	0	-	0	-	1	0	1	1
0	1	-	0	-	1	0	1	1

Table 16. Linear inequalities generated from f-TT$_{min}$

1	$x_1 + x_2 - x_3 + y_1 - y_2 + y_3 - p_1 + p_2 + 2 \geq 0$
2	$x_1 + x_2 - x_3 - y_1 + y_2 - y_3 - p_1 + p_2 + 3 \geq 0$
3	$-x_1 - x_2 - x_3 + y_1 - y_2 + y_3 - p_1 + p_2 + 4 \geq 0$
4	$-x_1 - x_2 + x_3 - y_1 - y_3 - p_1 + p_2 + 4 \geq 0$
5	$-x_1 - x_2 - y_1 - y_2 - y_3 - p_1 + p_2 + 5 \geq 0$
6	$x_1 + x_2 + x_3 + y_1 + y_2 + y_3 + p_1 + p_2 - 1 \geq 0$
7	$-x_1 + x_2 - y_1 + y_3 + p_1 - p_2 + 2 \geq 0$
8	$x_1 - x_2 - y_1 + y_3 + p_1 - p_2 + 2 \geq 0$
9	$-x_1 + x_2 + y_1 - y_3 + p_1 - p_2 + 2 \geq 0$
10	$x_1 - x_2 + y_1 - y_3 + p_1 - p_2 + 2 \geq 0$

References

1. Abdelkhalek, A., Sasaki, Y., Todo, Y., Tolba, M., Youssef, A.M.: MILP modeling for (large) S-boxes to optimize probability of differential characteristics. IACR Trans. Symmetric Cryptol. **2017**(4), 99–129 (2017). ISSN 2519-173X, https://doi.org/10.13154/tosc.v2017.i4.99-129
2. Biham, E.: New types of cryptanalytic attacks using related keys. J. Cryptol. **7**(4), 229–246 (1994). https://doi.org/10.1007/BF00203965
3. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_2
4. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like Cryptosystems. J. Cryptol. **4**, 3–72 (1991). Springer
5. Biham, E., Shamir, A.: Differential cryptanalysis of the full 16-round DES. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 487–496. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_34
6. Bogdanov, A.: Analysis and design of block cipher constructions. Ph.D. thesis (2009)
7. Bogdanov, A., et al.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74735-2_31
8. Boura, C., Coggia, D.: Efficient MILP modelings for S-boxes and linear layers of SPN ciphers. IACR Trans. Symmetric Cryptol. **3**, 327–361 (2020)
9. Beierle, C., et al.: The `SKINNY` family of block ciphers and its low-latency variant `MANTIS`. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_5
10. 'CryptoMiniSat5'. https://www.msoos.org/cryptominisat5
11. IBM ILOG: IBM ILOG CPLEX Optimization Studio V12.7.0 documentation (2016). Official webpage https://www-01.ibm.com/software/websphere/products/optimization/cplex-studio-community-edition/
12. Espresso Logic Minimizer. https://ptolemy.berkeley.edu/projects/embedded/pubs/downloads/espresso/

13. Gerault, D., Lafourcade, P., Minier, M., Solnon, C.: Revisiting AES related-key differential attacks with constraint programming. Cryptology ePrint Archive (2017)
14. Gohr, A.: Improving attacks on round-reduced Speck32/64 using deep learning. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 150–179. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_6
15. Gurobi Optimizer 7.5.2. https://www.gurobi.com
16. Hays, H.M.: A Tutorial on linear and differential cryptanalysis. Cryptologia **26**(3), 188–221 (2002)
17. Kim, H., Jeon, Y., Kim, G., Kim, J., Sim, B.-Y., Han, D.-G., Seo, H., Kim, S., Hong, S., Sung, J., Hong, D.: PIPO: a lightweight block cipher with efficient higher-order masking software implementations. In: Hong, D. (ed.) ICISC 2020. LNCS, vol. 12593, pp. 99–122. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-68890-5_6
18. Knudsen, L., Robshaw, M.J.B.: Block Cipher Companion. Springer, Heidelberg (2011). ISBN 978-3-642-17341-7. https://doi.org/10.1007/978-3-642-17342-4
19. Kumar, M., Suresh, T.S., Pal, S.K., Panigrahi, A.: Optimal differential trails in lightweight block ciphers ANU and PICO. Cryptologia **44**(1), 68–78 (2020)
20. Kumar, M., Yadav, T.: MILP based differential attack on round reduced WARP. In: Batina, L., Picek, S., Mondal, M. (eds.) SPACE 2021. LNCS, vol. 13162, pp. 42–59. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-95085-9_3
21. Kelsey, J., Schneier, B., Wagner, D.: Related-key cryptanalysis of 3-WAY, Biham-DES,CAST, DES-X, NewDES, RC2, and TEA. In: Han, Y., Okamoto, T., Qing, S. (eds.) ICICS 1997. LNCS, vol. 1334, pp. 233–246. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0028479
22. Logic Friday. https://sontrak.com/
23. Matsui, M.: On correlation between the order of S-boxes and the strength of DES. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 366–375. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0053451
24. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu, C.-K., Yung, M., Lin, D. (eds.) Inscrypt 2011. LNCS, vol. 7537, pp. 57–76. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34704-7_5
25. National Institute of Standards and Technology: Lightweight Cryptography, Finalists. NIST (2021). https://csrc.nist.gov/projects/lightweight-cryptography/finalists
26. National Institute of Standards and Technology: Federal Information Processing Standards Publication 197: Advanced Encryption Standard (AES). NIST (2001)
27. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 158–178. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_9
28. Sun, S., Hu, L., et al.: 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, Report 2014/747 (2014)
29. SAGE. https://www.sagemath.org/index.html
30. Sasaki, Yu., Todo, Y.: New differential bounds and division property of LILLIPUT: block cipher with extended generalized feistel network. In: Avanzi, R., Heys, H. (eds.) SAC 2016. LNCS, vol. 10532, pp. 264–283. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69453-5_15

31. Sasaki, Yu., Todo, Y.: New impossible differential search tool from design and cryptanalysis aspects. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10212, pp. 185–215. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_7
32. Sasaki, Yu., Todo, Y.: New algorithm for modeling S-box in MILP based differential and division trail search. In: Farshim, P., Simion, E. (eds.) SecITC 2017. LNCS, vol. 10543, pp. 150–165. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69284-5_11
33. Sun, S., et al.: Analysis of AES, SKINNY, and others with constraint programming. IACR Trans. Symmetric Cryptol. **1**, 281–306 (2017)
34. Yadav, T., Kumar, M.: Differential-ML distinguisher: machine learning based generic extension for differential cryptanalysis. In: Longa, P., Ràfols, C. (eds.) LATINCRYPT 2021. LNCS, vol. 12912, pp. 191–212. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88238-9_10
35. Zhu, B., Dong, X., Yu, H.: MILP-based differential attack on round-reduced GIFT. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 372–390. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_19