ON MEASURING SECURITY BOUNDS OF SOME CIPHERS USING MIXED INTEGER LINEAR PROGRAMMING (MILP) APPROACH

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

CAN TÜRESİN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CRYPTOGRAPHY

SEPTEMBER 2021

Approval of the thesis:

ON MEASURING SECURITY BOUNDS OF SOME CIPHERS USING MIXED INTEGER LINEAR PROGRAMMING (MILP) APPROACH

submitted by **CAN TÜRESİN** in partial fulfillment of the requirements for the degree of **Master of Science in Cryptography Department, Middle East Technical University** by,

Prof. Dr. A. Sevtap Selçuk Kestel
Director, Graduate School of **Applied Mathematics** _____

Prof. Dr. Ferruh Özbudak
Head of Department, **Cryptography** _____

Assoc. Prof. Dr. Ali Doğanaksoy
Supervisor, **Mathematics, METU** _____

Dr. Onur Koçak
Co-supervisor, **Tübitak Bilgem UEKAE** _____

Examining Committee Members:

Prof. Dr. Ferruh Özbudak
Mathematics Department, METU _____

Assoc. Prof. Dr. Ali Doğanaksoy
Mathematics Department, METU _____

Assoc. Prof. Dr. Oğuz Yayla
Institute of Applied Mathematics, METU _____

Assist. Prof. Dr. Ahmet Sınak
Mathematics and Computer Sciences Department,
Necmettin Erbakan University _____

Assist. Prof. Dr. Ayşe Nurdan Saran
Computer Engineering Department, Çankaya University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: CAN TÜRESİN

Signature :

ABSTRACT

ON MEASURING SECURITY BOUNDS OF SOME CIPHERS USING MIXED INTEGER LINEAR PROGRAMMING (MILP) APPROACH

Türesin, Can
M.S., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Ali Doğanaksoy
Co-Supervisor : Dr. Onur Koçak

September 2021, 39 pages

Block ciphers are one of the symmetric key encryption algorithms that are used in many devices. Its increasing popularity has led to the emergence of new cryptanalysis methods. Therefore, measuring block cipher's security bounds is one main indispensable need for its designers. Two of the most effective attacks on block ciphers are differential and linear cryptanalysis and these attacks' efficiencies are bonded with a number of active S-boxes of the cipher after a certain number of rounds. Consequently, measuring the number of active S-boxes is one of the techniques to measure the security bounds. There are various methods to calculate this number, both mathematically, and automatically. This thesis focuses on computing the minimum number of active S-boxes automatically, more specifically, using an optimization method called Mixed Integer Linear Programming, a method that achieved noticeable success across both the industrial and academic world. In this thesis, a summary of the studies in the literature on how to calculate the minimum number of active S-boxes with MILP will be given, the author's own ideas on this subject will also be shared, and the performance results of recent works and author's ideas will be compared.

Keywords: Cryptanalysis, Milp, Mixed Integer Linear Programming, Number of Active S-boxes

ÖZ

KARIŞIK TAMSAYILI DOĞRUSAL PROGRAMLAMA YAKLAŞIMI KULLANILARAK SP AĞI TABANLI ŞİFRELEME ALGORİTMALARININ GÜVENLİK LİMİTLERİNİN HESAPLANMASI ÜZERİNE

Türesin, Can
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali Doğanaksoy
Ortak Tez Yöneticisi : Dr. Onur Koçak

Eylül 2021, 39 sayfa

Blok şifreler, birçok cihazda kullanılan simetrik anahtar şifreleme algoritmalarından biridir. Artan popülaritesi, yeni kriptanaliz yöntemlerinin ortaya çıkmasına neden olmuştur. Bu nedenle, blok şifrelemenin güvenlik sınırlarını ölçmek, tasarımcıları için vazgeçilmez bir temel ihtiyaçtır. Blok şifrelere yönelik en etkili saldırılardan ikisi, diferansiyel ve doğrusal kriptanalizdir ve bu saldırıların etkinliği, belirli sayıda turdan sonra şifrenin aktif S-kutularının sayısı ile bağlantılıdır. Sonuç olarak, aktif S-kutularının sayısını ölçmek, güvenlik sınırlarını ölçmek için kullanılan tekniklerden biridir. Bu sayıyı hem matematiksel hem de otomatik olarak hesaplamak için çeşitli yöntemler vardır. Bu tez, hem endüstriyel hem de akademik dünyada gözle görülür bir başarı elde eden bir yöntem olan Karma Tamsayılı Doğrusal Programlama adlı bir optimizasyon yöntemini kullanarak minimum aktif S-kutu sayısını otomatik olarak hesaplamaya odaklanmaktadır. Bu tezde MILP ile minimum aktif s-box sayısının nasıl hesaplanacağına dair literatürdeki çalışmaların bir özeti verilecek, yazarın bu konudaki kendi fikirleri de paylaşılacak ve son dönemde yapılan çalışmaların performans sonuçları ve yazarın kendi fikirlerinin performans sonuçları karşılaştıralacaktır.

Anahtar Kelimeler: Kriptanaliz, Karışık Tamsayılı Doğrusal Programlama

ACKNOWLEDGMENTS

I would like to express my very great appreciation to my thesis supervisor Assoc. Prof. Dr. Ali Doğanaksoy for his patient guidance, enthusiastic encouragement and valuable advices during the development and preparation of this thesis. His willingness to give his time and to share his experiences has brightened my path. I would also like to thank Assoc. Prof. Dr. Zülfükar Saygı who has guided me to start working on this subject. I would also like to express my gratitude to Murat Burhan İlter, whom I consulted on every issue when I started working on Mixed Integer Linear Programming. I also want to thank my co-advisor Dr. Onur Koçak for all of his support and guidance, he always helped me in clearing my way through obstacles. when I wanted to consult someone, he was always there. I am also grateful to my colleagues and to my friends for their support during this time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

SPN	Substitution-Permutation Network
AES	Advanced Encryption Standard
MILP	Mixed Integer Linear Programming
S-box	Substitution-box
imp	Impossible Path
pos	Possible Path
H_{set}	Convex Hull Set of Possible Paths
X_{set}	Set of Impossible Paths
C.H.	Convex Hull
DBN	Differential Branch Number

CHAPTER 1

INTRODUCTION

Block Ciphers are symmetric key encryption algorithms that are widely used in many electronic devices. SPN (Substitution-Permutation Network) structures are one of the main structures that block ciphers use, and block ciphers with these structures are known to have an easy and effective implementation in terms of hardware and software. As the name suggests, these structures consist of substitution and permutation layers. New attacks to recover the private key are developed after the escalating use of block ciphers in information security. The two main and effective attacks known are differential [4], and linear [16] cryptanalysis. These attacks exploit cases that occur with high probability and hence recover the keys. These cases mainly occur on the substitution layer of ciphers, i.e S-boxes. Therefore, not only for the attacker but also for the designers, it is important to count the minimum number of active S-boxes after a certain number of rounds are completed to say that the cipher is provable secure.

The designers of AES (Advanced Encryption Standard) introduced the Wide Trail Strategy [10] in 2002, which is a designing strategy to resist differential and linear cryptanalysis. Resistance against these two attacks are one of the most important requirements for a newly designed block cipher.

Along with the mathematical approach to calculate the minimum number of active S-boxes, another approach to calculate this number automatically has been made by Aoki et al. [3].

Computing the minimum number of active S-boxes with Mixed integer linear programming (MILP) was another technique for the second approach used in work done

by Mouha et al. In [17]. MILP is an optimization technique used to maximize or minimize an objective function under certain circumstances. MILP became popular among cryptography recent decades. We suggest reader to read [5], [26], [2], [15], [14], [13], [12], [19], in order to investigate the other design and analysis areas that is used. These papers are out of our scope as we investigate the number of active S-boxes after certain rounds. The objective function here is to reduce the number of active S-boxes, and the constraints are equations that are obtained by difference propagation of the operations across cipher. Therefore, their work covers constructing the MILP models of the operations within the cipher considering these situations. It consists how to construct the models of XOR and linear transformation operations using diferential branch number.

After the work done by Mouha et al., Sun et al. made some extensions to it in [20] to compute the minimum number of active S-boxes of SPN-based block ciphers. In their work, shortcomings of Mouha's technique are examined, and how these shortcomings can be overcome are mentioned. One year later, in the works [21], and [22], Sun et al. describes how to refine the S-box modeling of their previous technique for ciphers. This refining process can be done by finding the convex hull (convex hull of a set A, smallest set that contains A) of the set of possible input-output difference pairs of the S-boxes and then using the inequalities that define this convex hull set. However, using the initial set of these inequalities is simply too large to add to MILP model. Therefore, they define a greedy algorithm to reduce number the constraints to construct that refined S-box model. Thanks to this greedy algorithm, the number of inequalities defining the S-box operation has been significantly reduced.

In 2017, Sasaki et al. [18] described another technique to reduce the set of convex hull inequalities instead of Sun et al's greedy algorithm. Their technique constructs a MILP model from initial set of convex hull inequalities, and impossible input-output difference pairs of an S-box. Objective function of model here is minimizing the set of convex hull inequalities, and constraints are constructed with the impossible difference pairs and convex hull inequalities that eliminates them. A slightly better reduction is achieved compared to results of the greedy algorithm by means of this MILP modeling.

In 2020, Boura et al.' work [8] updated the reduction process of the number of convex hull inequalities of possible input-output difference pairs of S-boxes. Since Sasaki et al.'s reduction via MILP is the most effective method known so far, using this method, it aims to find a more reduced result by first making an expansion to the set of convex hull inequalities to be reduced. Their expansion done by the taking sum of some convex hull inequalities that satisfies certain properties. In this way, Their work achieved even better results than Sasaki et al.'s MILP reduction method.

1.1 Our Contribution

We tried to find a different set of inequalities describing an S-box other than the work done by Boura et al. [8]. To achieve this, we categorized impossible input-output difference pairs of an S-box and extract subsets of inequalities that can eliminate each category. In this way we were able to find a new set of inequalities describing an S-box. We applied our idea on 4×4 S-boxes of SPN-based block ciphers Present [6], Prince [7], and Klein [11], constructed models according to Sasaki et al.'s work, Boura et al.'s work and our work. We had able to achieve a better performance to find minimum number of active S-boxes for the ciphers Present, Prince, and Klein. The results are given in Table 3.9 at Chapter 3.

1.2 Outline of the Thesis

Chapter 2 investigates the works done by Mouha et al. [17], Sun et al. [20],[21],[22], Sasaki et al. [18], and Boura et al. [8]. It is explained how MILP modeling can be done by utilizing difference propagation, structural properties of encryption operations (S-boxes specifically).

Chapter 3 covers our contribution to reduce the number of constraints to model an S-box operation. We categorized impossible input-output difference pairs of an S-box according to their certain properties, and then obtained a new set of constraints other than the related works mentioned in chapter 2 to eliminate them. Then the MILP models from recent techniques and our adjustments are compared for the block ci-

phers Present [6], Prince [7], and Klein [11]. A slightly improvement in performance is observed for the models of these ciphers. These results are given in Table 3.9.

Chapter 4 refers to the importance of calculating the security bounds of block ciphers. It summarizes mentions that the minimum number of active s-boxes in certain rounds is also one of the security parameters, and it also mentioned that how this number is calculated and how to calculate it with MILP. It briefly touches upon what we have done in this regard and tells about the work we will do as a future work.

1.3 Preliminaries

Mixed Integer Linear Programming is a method used to optimize a particular problem under related constraints. It is a branch of linear programming, which differs as it uses both integers and real numbers intending to boost the overall performance. The problem is defined as minimization or maximization of a specific function according to the situation, called the objective function. The objective function is optimized under certain circumstances. These circumstances are called constraints, need to be linear inequalities. This optimization method achieved a noticeable performance in both the industrial and academic worlds.

Mixed Integer Linear Programming can be defined as follows:

Definition 1. *Find a vector* $x \in Z^k \times R^{n-k} \subseteq R^n$ *with* $Ax \leq b$*, such that the linear function* $c_1x_1 + c_2x_2 + ... + c_nx_n$ *is minimized (or maximized), where* $(c_1, ...c_2) \in R^n$*,* $A \in R^n$*,* $A \in R^{m \times n}$*, and* $b \in R^m$*.*

Aside from being used a lot to optimize the challenging mathematical problems used in cryptography, such as the 0-1 knapsack problem and traveling salesman problem, it has been used in cryptanalysis in recent years [17].

Example: (Furniture Factory Problem)

Assume that there is a furniture production factory that produces tables and chairs. Let p_1 be the number of chairs made, and p_2 be the number of tables produced. The products require wood and labor to be built, and they have different price tags. As-

sume that a chair needs 10 units of wood, 12 hours of labor work, and its price tag is 50 dollars. On the other hand, let a table needs 25 units of wood, 18 hours of labor work, and let its price tag is 95 dollars. The factory has capacities for both blocks of wood and the labors 500 units and 650 hours, respectively. The problem is, how many chairs and tables must be produced in order to maximize the profit?

Table 1.1: Resources required for crafting the Products

Data	Chair	Table	Capacity
Wood	10	25	500
Labour	12	18	650
Price Tag	50	95	–

Our objective function to be maximized here is the sum of prices of the products, so that it is $50p_1 + 95p_2$. Also the number of produced products cannot be negative, therefore we have the constraint $p_1, p_2 \geq 0$. Moreover, we need to include resource constraints according to their capacities, which are $10p_1 + 25p_2 \leq 500$ and $12p_1 + 18p_2 \leq 650$.

All in all, the furniture factory problem can be modeled like this:

$$\begin{cases} \textbf{maximize} \\ 50\mathrm{p}_1 + 95p_2 \\ \textbf{subject to} \\ 10\mathrm{p}_1 + 25p_2 \leq 500 \\ 12\mathrm{p}_1 + 18p_2 \leq 650 \\ \mathrm{p}_1, p_2 \geq 0 \end{cases}$$

MILP can be used to calculate the minimum active S-boxes for n rounds of ciphers, and in this thesis, the recent research that has been conducted according to this problem will be investigated. Our objective function is the sum of active S-boxes after the n rounds, so that as one needs to minimize this number, we will aim to minimize this objective function. We will construct our constraints with the operations that have been done to the bits. Therefore, we need to consider all the confusion and diffusion layer bit operations to cover the cipher fully.

On the other hand, as MILP calculates the minimum active S-box count for a cipher,

it also calculates the input difference necessary to achieve this S-box count. One can infer that it can also be used to calculate the best pattern, i.e, the distinguisher for differential cryptanalysis. Since it is difficult to find a distinguisher for a specific cipher, it can be practical for the analyst.

CHAPTER 2

CALCULATING MINIMUM NUMBER OF ACTIVE S-BOXES USING MILP

This chapter will examine various works related to calculating the minimum number of active S-boxes using MILP.

2.1 Introduction

As it aims to find the best possible solution to a particular problem, MILP is used to optimize the problems covering the cryptanalysis side of cryptography. In 2011, Mouha et al. published a paper to show how to compute the number of active S-boxes for the first time by using MILP in order to reduce the workload for designers and cryptanalysts.

Differential and linear cryptanalysis are statistical attacks. Their primary purpose is to find a distinguisher that is a pattern with high probability. Using this distinguisher, check if the attacker can find the key used in cipher in polynomial time. In order to do this, analysts should scan all possible patterns and find the pattern with the highest probability, which is a time-consuming problem as there are so many options. Also, there is the possibility of human errors. Mouha et al. state that if one can correctly model the cipher for MILP, it can find the best possible pattern along with the number of active S-boxes.

2.2 Constructing MILP Model of Ciphers

In their work, Mouha et al. construct the models of the ciphers Enocoro-128V2 and AES. The concept of differential branch number must be mentioned to define the difference propagation in the ciphers correctly. The differential branch number is the minimum number of 1s in input and output differences in an operation, which is eventually used for a reasonable explanation about the propagation. More Formally, the differential branch number for an S-box can be defined as:

Definition 2. *Differential branch number of S-box B_s is defined as,*

$$B_s = min\{wt((\alpha \oplus \beta) \| (S(\alpha) \oplus S(\beta))) : \alpha, \beta \in \mathbb{F}_2^w\}$$

where $\alpha \neq \beta$.

We will cover how to construct MILP model of a cipher to calculate the minimum number of active S-boxes on stream cipher Enocoro-128v2. Therefore, its description must be given:

Description of Enocoro-128v2

Enocoro-128v2 is a stream stream cipher proposed in 2010 by Hitachi [25] which was inspired by Panama construction [9]. In its internal state, it has a buffer b which is 32 bytes long $(b_0, ..., b_{31})$ and it also has a state variable a which is 2 bytes long (a_0, a_1). The state update of Enocoro-128v2 is given in Figure 2.1.

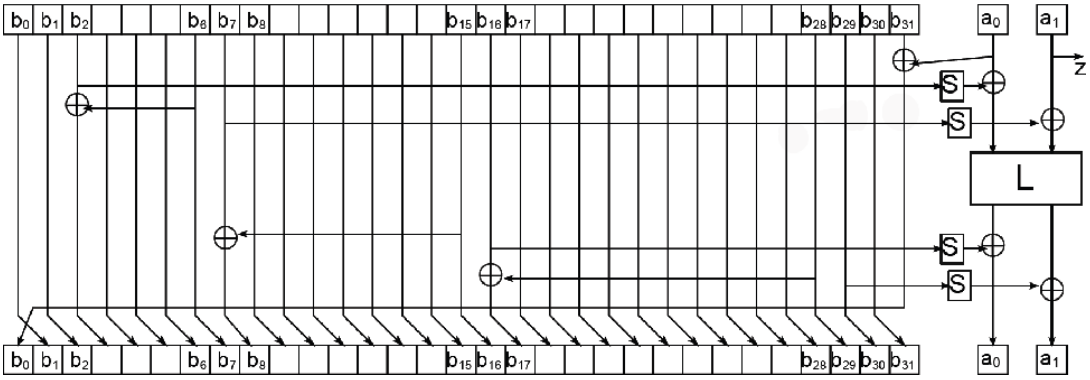

Figure 2.1: State update in enocoro128v2

The state function uses the functions ρ and λ to update the internal state.

Function ρ uses an 8-bit S-box, the Linear transformation L, and XOR operations to update the state variable a. Here L is a linear transformation that is defined in $GF(2^8)$ and can be expressed as:

$$\begin{bmatrix} v_0 \\ v_1 \end{bmatrix} = L(u_0, u_1) = \begin{bmatrix} 1 & 1 \\ 1 & d \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ u_1 \end{bmatrix} \tag{2.1}$$

where $d \in GF(2^8)$.

Here $d = 0x02$, the two inputs of L are $u_0 = a_0^t \oplus S[b_2^t]$ and $u_1 = a_1^t \oplus S[b_7^t]$. The outputs of L are used to calculate the update state as:

$$a_0^{t+1} = v_0 \oplus S[b_{16}^t],$$
$$a_1^{t+1} = v_0 \oplus S[b_{29}^t].$$

Function λ uses XORs and bytewise rotation in order to update the buffer b. This function can be simply expressed as:

$$b_i^{t+1} = \begin{cases} b_{31}^t \oplus a_0^t, \text{ if } i = 0, \\ b_2^t \oplus b_6^t, \text{ if } i = 3, \\ b_7^t \oplus b_{15}^t, \text{ if } i = 8, \\ b_{16}^t \oplus b_{28}^t, \text{ if } i = 17, \\ b_{i-1}^t, \text{ if } i = \text{otherwise.} \end{cases}$$

Constructing the MILP Model of Enocoro-128v2

As shown in the figure 2.1, the update function of the stream cipher has 8 XOR operators, 1 linear transformation L, and 4 S-box operations. In order to model the ciphers and find their security bounds, i.e calculate the number of active S-boxes in specified rounds, one should correctly model its XOR operations and the linear transformation L.

2.2.1 MILP Model of XOR

To correctly model the differential propagation for XOR modeling, we refer to definition 2 (Differential Branch Number). According to its definition, one should check all

the operation's non-zero input and output combinations to find its differential branch number. All possible non-zero inputs and their outputs suggest that the differential number of XOR operation is 2. This result means that the sum of the weights of inputs and the output must be at least two. To construct the XOR model, let x_1 and x_2 be the two inputs, and y is the output, i.e $x_1 \oplus x_2 = y$. This XOR operation can be expressed as the following four inequalities:

$$\begin{aligned} x_1 + x_2 + y &\geq 2d, \\ d &\geq x_1, \\ d &\geq x_2, \\ d &\geq y, \end{aligned}$$

where d is a dummy binary variable where it is equal to 0 if all of the input and output variables are zero, and 1 otherwise. The story behind why Mouha et al. used four inequalities for the linear representation is to express the propagation correctly. Let us examine all cases. First, assume that all the input and output variables are zero. Then, the dummy variable d is also zero, and so the second, third and fourth inequalities hold. Moreover, as both sides of the first equation are zero, it also holds. Secondly, assume that one of the inputs is zero and the other is non-zero, without loss of generality, say x_1 is non-zero, x_0 is zero. Dummy variable d is 1 in this case, and so the LHS of the first inequality must be greater than 2. Therefore there needs to be at least one bit active, i.e., y must be non-zero, which is the nature of an XOR operation. Finally, consider both of the input variables are non-zero. Then of course dummy variable d is 1, and as x_1 are x_2 are non-zero the LHS is always greater than 2.

2.2.2 MILP Model of Linear Transformation

The structure of linear transformation L is given in Equation 2.1. Since the minimum total weight of its all possible non-zero inputs and outputs is 3, its differential branch number is 3. Therefore, one can express this operation like the previous XOR modeling, with the constraints which are given below:

$$x_1 + x_2 + y_1 + y_2 \geq 3d,$$
$$d \geq x_1,$$
$$d \geq x_2,$$
$$d \geq y_1,$$
$$d \geq y_2,$$

where d is again a dummy variable which is 0 if every input and output variables are zero, and 1 otherwise.

2.2.3 Applying Models for Enocoro-128v2

Enocoro description shows that it has eight XOR operations in its structure. Let us call the initial difference vectors as $\{x_0, \ldots, x_{31}\}$ for the buffer b and $\{x_{32}, x_{33}\}$ for state variable a. The first XOR operation takes x_{31} and x_{32} as inputs and let x_{33} be a new difference vector that is output of this operation. Then, from Subsection 2.2.1, one can construct the model of this XOR operation as:

$$x_{31} + x_{32} + x_{34} \geq 2d_0,$$
$$d_0 \geq x_{31},$$
$$d_0 \geq x_{32},$$
$$d_0 \geq x_{34}.$$

Since one uses an exact way to model the other seven XOR operations, let us define it in the following table.

Table 2.1: Difference variables for XOR operations of Enocoro-128v2.v_0 and v_1 are the outputs of the Linear Transformation L.

Input 1	Input 2	Output
x_{31}	x_{32}	x_{34}
x_2	x_{32}	x_{35}
x_2	x_6	x_{36}
x_7	x_{33}	x_{37}
x_7	x_{15}	x_{38}
x_{16}	v_0	x_{39}
x_{16}	x_{28}	x_{40}
x_{29}	v_1	x_{41}

From Table 2.1, one can construct the XOR models with the desired variables.

On the other hand, the inputs for the linear transformation L are the outputs of the second and fourth XOR operations, namely x_{35} and x_{37}, respectively. Let us call the output difference variables of this operation as x_{42} and x_{43}. From subsection 2.2.2, one can construct the model as:

$$
\begin{aligned}
x_{35} + x_{37} + x_{42} + x_{43} &\geq 3d_8, \\
d_8 &\geq x_{35}, \\
d_8 &\geq x_{37}, \\
d_8 &\geq x_{42}, \\
d_8 &\geq x_{43}.
\end{aligned}
$$

After constructing the L model, one can call the previous differential variables v_0, v_1 which were the inputs for sixth and eighth XOR operations, as x_{42} and x_{43} respectively.

2.2.4 Computing Security Bounds of a Cipher

In the subsection 2.2.3, it was explained how to apply the MILP models of XOR and linear transformation in a specific cipher. It is seen that how the difference vectors are related to each other, and how the new output difference variables are generated. To analyze the model in a MILP solver, the analyzer now needs to have an objective

function to be maximized or minimized. As mentioned in section 2.1, the objective function here is to find the minimum number of active S-boxes. Before moving on, it would be meaningful to define an active S-box.

Definition 3. *Let S be and $n \times n$ S-box and $\{x_0, \ldots, x_{n-1}\}$ be the input difference vector of S. Then S is called **active** if any of this difference variables are 1 and S is not active if $x_i = 0$, $\forall i \in \{0, \ldots, n-1\}$.*

Definition 3 suggests that to investigate the activeness of an S-box, one should check its input difference variables. Therefore, e.g., back to Enocoro whose structure is given in Figure 2.1, to check whether the first S-box is active in the first round, one should look for the difference variable x_2. Similarly, x_7, x_{16} and x_{29} for the second, third, and fourth S-boxes, and since the analyzer desires to find a path for minimum possible active S-box case, they should minimize the sum of this variables. Then, for the 1 round Enocoro-128v2, the related objective function to be minimized becomes:

$$\text{Objective Function: } \textbf{Minimize } x_2 + x_7 + x_{16} + x_{29}.$$

2.3 Modeling SPN Ciphers To Analyze With MILP

In 2013, inspired by the work mentioned in the previous chapter, i.e Mouha et al.'s work, Sun et al. published a paper regarding a further study to use MILP in cryptanalysis [20]. Their work investigated the SPN-based block cipher PRESENT [6] and defined some new structures for the S-box modeling.

Back to Mouha et al.'s work, it was not enough to fully cover the diffusion effect of bitwise permutation since their work was for word-level permutation layers. Therefore, Sun et al., extended Mouha et al.'s work, introduced new modeling to SPN structures for XOR operations, and applied these operations to the international standard for lightweight symmetric key cryptography, PRESENT, and showed the ciphers security bounds according to their model.

2.3.1 Extension Done by Sun et al.

To introduce the work done by Sun et al., let us call an r round SPN block cipher with n bit state and its $s \times s$ S-box as SPN(n, s, r). For example, we can call PRESENT as PRESENT$(64, 4, 31)$.

2.3.2 Representing MILP Variables for the Structure

Consider $\{x_{r\_0}, x_{r\_1}, \ldots, x_{r\_n-2}, x_{r\_n-1}\}$ be the round state difference vector of a given SPN cipher at rth round. Then, for any x_{r_i} where $i \in \{0, n-1\}$,

$$x_{r\_i}= \begin{cases} 1, \text{ if there is a difference in that bit position, i.e bit is active,} \\ 0, \text{ otherwise.} \end{cases}$$

Moreover, Consider $\{A_{r\_0}, x_{r\_1}, \ldots, A_{r\_\frac{n}{s}-2}, A_{r\_\frac{n}{s}-1}\}$ be S-boxes of a given SPN cipher at rth round. Then, for any A_{r_i} where $i \in \{0, \frac{n}{s}-1\}$,

$$A_{r\_i}= \begin{cases} 1, \text{ if the } S\text{-box } A_{r\_i} \text{ is active,} \\ 0, \text{ otherwise.} \end{cases}$$

2.3.3 Modeling Input Output Bits of an S-box According to Differential Branch Number

In this subsection, using the differential branch number $\mathbb{B}$ of the cipher's S-box, we will show how to connect the input and output bits to each other. For an $s \times s$ S-box S, define $\{x_0, x_1, \ldots, x_{s-2}, x_{s-1}\}$ and $\{y_0, y_1, \ldots, y_{s-2}, y_{s-1}\}$ as input and output bits, respectively. Inspired by Mouha et al. work on XOR modeling in subsection 2.2.1, one can model this as:

$$\begin{gathered} x_0 + x_1 + \cdots + x_{s-2} + x_{s-1} + y_0 + y_1 + \cdots + y_{s-2} + y_{s-1} \geq \mathbb{B} \cdot d, \\ d \geq x_0, \\ \vdots \end{gathered}$$

$$
\begin{aligned}
d &\geq x_{s-1}, \\
d &\geq y_0, \\
&\vdots \\
d &\geq y_{s-1},
\end{aligned}
$$

2.3.4 Connecting a State's Bit Variables With S-box Variables

Now our goal here is, to connect the variables that have been given in previous subsection. To achieve this, let us define input and output difference vectors of an S-box A_{r_i} as $\{x_{r\_0}, x_{r\_1}, \ldots, x_{r\_s-2}, x_{r\_s-1}\}$ and $\{y_{r\_0}, y_{r\_1}, \ldots, y_{r\_s-2}, y_{r\_s-1}\}$, respectively. The definition of S-box suggests that its bijective, so that a non-zero input always maps to a non-zero output, which makes it clear that a non-zero input or output bit always turns the S-box to active. This condition can be achieved in a MILP model with the following constraints:

$$
\begin{cases}
x_{r\_0} - A_{r\_i} \leq 0, \\
x_{r\_1} - A_{r\_i} \leq 0. \\
\vdots \\
x_{r\_s-1} - A_{r\_i} \leq 0, \\
y_{r\_0} - A_{r\_i} \leq 0, \\
y_{r\_1} - A_{r\_i} \leq 0, \\
\vdots \\
y_{r\_s-1} - A_{r\_i} \leq 0,
\end{cases}
$$

These constraints indicate that A_{r_i} is active if one of the variables is non-zero. Moreover, because of the same rule, the other side of the medallion must be achieved, namely, if the S-box is active, at least one of the difference variables must be non-zero. Therefore, the following constraint must also be added:

$$x_{r\_0} + x_{r\_1} + \cdots + x_{r\_s-2} + x_{r\_s-1} + y_{r\_0} + y_{r\_1} + \cdots + y_{r\_s-2} + y_{r\_s-1} - A_{r\_i} \geq 0$$

Up to this point, the constraints are similar to the ones that Mouha et al. defined in

their work for XOR operation. However, these are still not enough to fully cover the bijection property of the S-box operation. Consider the following counterexample for this assertion:

Assume that $\{x_0, x_1, \ldots, x_{s-2}, x_{s-1}\}$ and $\{y_0, y_1, \ldots, y_{s-2}, y_{s-1}\}$ be the input and output difference vectors. Assume differential branch number $\mathbb{B}$ is less than S-box size s. Let $\mathbb{B}$ of input variables are non-zero and all the other variables are zero. Without lost of generality, say $\{x_0, \ldots, x_{\mathbb{B}}\}$ is 1. As shown in subsection 2.3.3, the dummy variable d is need to be 1 as there are non-zero variables and so the constraints $d \geq x_i$ and $d \geq y_i$ for all $i \in \{0, s-1\}$. For the constraint,

$$x_0 + x_1 + \cdots + x_{s-2} + x_{s-1} + y_0 + y_1 + \cdots + y_{s-2} + y_{s-1} \geq \mathbb{B} \cdot d,$$

the right hand side is;

$$\mathbb{B} \cdot d = \mathbb{B} \cdot 1 = \mathbb{B}.$$

Variously, the left hand side is;

$$\begin{aligned} x_0 + x_1 + \cdots + x_{\mathbb{B}} + \cdots + x_{s-2} + x_{s-1} + y_0 + y_1 + \cdots + y_{s-2} + y_{s-1} = \\ = \underbrace{1 + \cdots + 1}_{\mathbb{B} \text{ times}} + \underbrace{0 + \cdots + 0}_{2 \cdot s - \mathbb{B} \text{ times}} = \mathbb{B}. \end{aligned}$$

Therefore, as both the LHS and the RHS are $\mathbb{B}$, the constraint holds, so that this example does not violate our constraints. However, this a counterexample to the bijection property of the S-box as the input is non-zero, but the output is zero. Therefore, there need to be extra constraints in order to correct this problem. Sun et al. suggested two new constraints to overcome this situation, which are:

$$\begin{aligned} s \cdot (x_0 + \cdots + x_{s-1}) - (y_0 + \cdots + y_{s-1}) \geq 0, \\ s \cdot (y_0 + \cdots + y_{s-1}) - (x_0 + \cdots + x_{s-1}) \geq 0. \end{aligned}$$

The reason to multiply the positive variables with S-box size s is to satisfy necessary active negative bits with the less active positive variables as an S-box can map one active input variable to many output variables and similarly many to one.

2.3.5 Cutting Off Impossible S-box Paths

From subsections 2.3.3 and 2.3.4, one can model S-box of a cipher according to its branch number, and its size. However, constraints derived from these attributes are not enough to eliminate all of the impossible paths of the cipher. Here, the definition of a differential path must be given in order to understand the cutting off concept better:

Definition 4. *Let S be an $s \times s$ S-box, $\{x_0, \ldots, x_{s-1}\}$ and $\{y_0, \ldots, y_{s-1}\}$ be the input and output differences, respectively. Then, $\{x_0, \ldots, x_{s-1}, y_0, \ldots, y_{s-1}\}$ is called a **differential path**. When an input difference $\{x_0, \ldots, x_{s-1}\}$ leads to an output difference $\{y_0, \ldots, y_{s-1}\}$ with a non-zero probability, then this differential path is called **possible**. Otherwise, it is called **impossible**.*

The impossible and possible paths of an S-box are unique to their own so that if one creates a model of an S-box without considering these paths, the model could be very superficial. Moreover, all the S-boxes with the same size and branch number would have the same solution space, which is inaccurate as they have different mappings. To model S-boxes more accurately, considering possible and impossible paths, Sun et al. [22] suggested new constraints, which are called **cutting off inequalities**.

Definition 5. ***Cutting off inequalities** of an S-box are the linear inequalities that are satisfied by all possible paths of the S-box, but violated by at least one impossible path of the S-box.*

An accurate model can be constructed if one creates a sufficient number of cutting off inequalities to eliminate all impossible paths of an S-box. Therefore, the problem now becomes how to create enough cutting off inequalities.

2.3.5.1 Generating Cutting off Inequalities

In this part, we will see how to generate the cutting off inequalities. Sun et al. suggest two methods for that, logical condition modeling and convex hull modeling [22].

Logical Condition Method To Construct Cutting off Inequalities

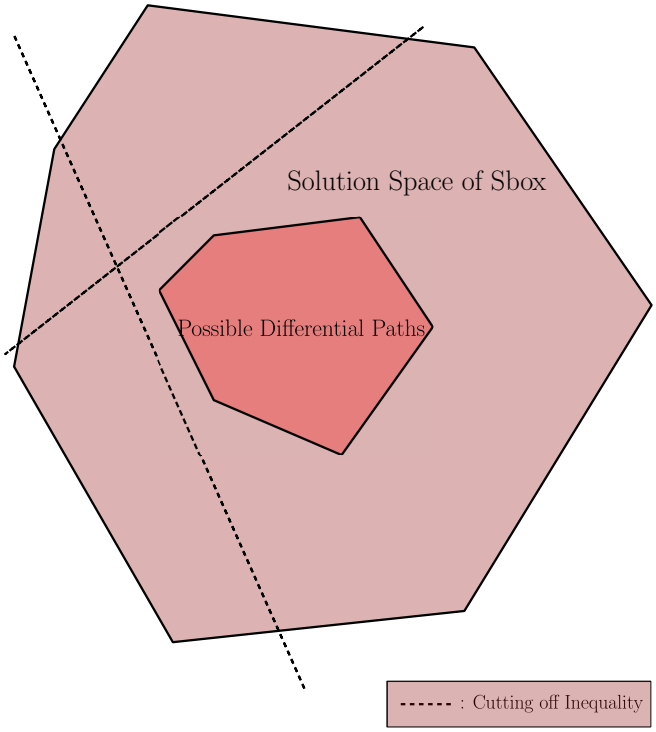

Figure 2.2: Cutting Off Inequalities of an S-box

This method aims to exploit from characteristics of certain input or output bits in an S-box. The definition for these individual bits is given below:

Definition 6. *For a specific S-box input difference Δx, it can be observed that some of the bits of the corresponding output difference can be unchanged. Similarly, this anomaly can also be observed in the input difference bits for a specific output difference Δy. This constant bits are called* ***undisturbed bits****[24].*

This modeling approach constructs inequalities of logical conditions collected from undisturbed bits of an S-box to eliminate the impossible paths in its solution space. A formal definition for the logical condition is given below:

Definition 7. *Let $\{x_0, \ldots, x_{s-1}\}$ be the input bit differences of an $s \times s$ an S-box whose 0-1 results are $\{\rho_0, \ldots, \rho_{s-1}\}$ and $\{y_0, \ldots, y_{s-1}\}$ be the output difference bits corresponding to this input. Let $\{y_n, \ldots, y_k\}$ be the undisturbed output bit differences such that $n \leq k$ and $0 \leq n$, $k \leq s$, whose 0-1 results are $\{\sigma_n, \ldots, \sigma_k\}$. Then, one can show this case as the logical condition:*

$$\{x_0, \ldots, x_{s-1}\} = \{\rho_0, \ldots, \rho_{s-1}\} \Longrightarrow \{y_n, \ldots, y_k\} = \{\sigma_n, \ldots, \sigma_k\}$$

Considering the definition and variables defined in the definition, the formula to construct logical condition inequality is given below:

$$\sum_{i=0}^{s-1}(-1)^{\rho_i}x_i + \sum_{i=n}^{k}(-1)^{\sigma_i+1}y_i - \sum_{i=n}^{k}\sigma_i + \sum_{i=0}^{s-1}\rho_i \geq 0$$

Using undisturbed bits, one can construct the logical condition inequalities based on the corresponding input or output differences.

To making things more clear, consider four propositions on Present's S-box, given in section 4 of Tezcan's work [24]:

1. If the input difference is 1001, then the least significant bit in output is an undisturbed bit, which is always 0.
2. If the input difference is 0001 or 1000, then the least significant bit in output is an undisturbed bit, which is always 1.
3. If the output difference is 0001 or 0100, then the least significant bit in input is an undisturbed bit, which is always 1.
4. If the output difference is 0101, the least significant bit in input is an undisturbed bit, which is always 0.

Taking into account these propositions, logical condition constraints for Present's S-box can be constructed from the formula above:

$$\begin{gathered}
-x_0 + x_1 + x_2 - x_3 - y_3 + 2 \geq 0, \text{from proposition } \mathbf{1}, \\
x_0 + x_1 + x_2 - x_3 + y_3 \geq 0, \text{from proposition } \mathbf{2}, \\
-x_0 + x_1 + x_2 + x_3 + y_3 \geq 0, \text{from proposition } \mathbf{2}, \\
x_3 + y_0 + y_1 + y_2 - y_3 \geq 0, \text{from proposition } \mathbf{3}, \\
x_3 + y_0 - y_1 + y_2 + y_3 \geq 0, \text{from proposition } \mathbf{3}, \\
-x_3 + y_0 - y_1 + y_2 - y_3 + 2 \geq 0, \text{from proposition } \mathbf{4},
\end{gathered}$$

Although these constraints eliminate the impossible paths conflicting with the value of undisturbed bits, this method lacks to eliminate all impossible paths of an S-box.

Convex Hull Method To Construct Cutting off Inequalities

Convex Hull of a set of points A is the most miniature convex set (for any two points in the set, the line segment joining them is also in the set) that contains A.

The problem here is to divide the solution space of an S-box into two parts; possible and impossible paths. If one can find the convex hull of the possible paths, then the S-box can be modeled more accurately. The inequalities defining the convex hull of the possible paths will be called the cutting off inequalities that we defined in definition 5.

Fortunately, an open-source mathematical software system called **SageMath** has its built-in Convex hull inequality extraction method in its **sage. geometry. polyhedron** library. The method is called $inequality\ generator()$, and one needs first to define a polyhedron with the vertices selected from possible paths of the S-box, then the method returns the inequalities whose solution space gives the S-box's convex hull.

However, the inequalities collected from SageMath's method are too many to apply in MILP modeling. For example, for the S-box of block cipher Prince, there are 300 cutting off inequalities to eliminate its impossible paths. Moreover, since its state is 64-bit and S-box is 4-bit, the S-box layer works 16 times to substitute the state entirely, and it has 12 S-box operations in total (although it has 11 rounds, there are 2 S-box operations in its middle round). Therefore, in total, it makes $300 \cdot 16 \cdot 12 = 57600$ constraints to be added to its MILP model in order to eliminate every impossible path in its encryption process, which is overwhelmingly costly.

In their work, Sun et al. [22] observed that there are some ineffective inequalities in convex hull of possibles, in terms of their eliminating impossible sets. In other words, the impossible sets of some convex hull inequalities are subsets of union of some other convex hull inequalities' elimination sets. For example, let z_i, z_j, z_k be cutting off inequalities in the convex hull set $\{z_0, z_1, \ldots z_n\}$ which eliminates the impossible paths $\{p_0, p_1, \ldots, p_k\}$ where $i, j, k \leq n$. Assume that z_i eliminates the impossible subset $E_i = \{p_1, p_5, p_7\}$, while z_j eliminates $E_j = \{p_1, p_8, p_9\}$ and z_k eliminates $E_k = \{p_1, p_3, p_5, p_7\}$. It is obvious that $E_i \subset E_j \cup E_k$. Therefore there is no need to use z_i as a cutting of inequality as its eliminated impossible subset is can be eliminated by union of other cutting off inequalities.

Sun et al. [22] have made a greedy approach to select the best cutting off inequalities through a set of the convex hull for a given output size as an input. In this approach, the set of convex hull inequalities H_{set}, the set of impossible paths X_{set} for a specified S-box, and the number n, which is the number of cutting off inequalities to be selected from H_{set} are the inputs of the algorithm. The algorithm loops n times, and in every iteration, it omits the convex hull inequality, which eliminates the most impossible paths, adds it to the output set, and then moves to the next loop. The pseudocode of greedy algorithm follows as:

Algorithm 1 Greedy Algorithm to select k inequalities from Convex Hull set

Input: positive integer k, Convex Hull H_{set}, set of impossible paths X_{set}

Output: the set of k cutting of inequalities $Result$

```
index ← 0
for j from 0 to k do
    for i in H_set do
        E ← E + {The set of impossibles eliminated by i}
    end for
    index ← index of set in E with max number of elements
    result ← result + H_set[index]
    for imp in E[index] do
        X_set ← X_set − imp
    end for
    H_set ← H_set − H_set[index]
end for
return result
```

The problem with the greedy algorithm in Sun et al.'s work [22] is, although it makes a smaller solution space via n selected cutting off inequalities, it may not eliminate the whole impossible path set if one gives the input n less than the minimum required. To overcome this problem, Sun et al. updated their algorithm to check if the selected n inequalities eliminate the impossible set entirely. In this scenario, the algorithm does not require the cutting inequality count n as an input, and it just continues to run until the output set eliminates the whole impossible paths. The pseudocode of the

updated algorithm follows as:

Algorithm 2 Greedy Algorithm to select cutting off inequalities from Convex Hull set for an S-box which completely eliminates the whole impossible paths

Input: Convex Hull H_{set}, Set of impossible paths X_{set}

Output: the set of cutting off inequalities $result$

$index \leftarrow 0$
while $true$ **do**
for i **in** H_{set} **do**
    $E \leftarrow E + \{$The set of impossibles eliminated by i$\}$
end for
$index \leftarrow$ index of set in E with max number of elements
$result \leftarrow result + H_{set}[index]$
for imp **in** $E[index]$ **do**
    $X_{set} \leftarrow X_{set} - imp$
end for
$H_{set} \leftarrow H_{set} - H_{set}[index]$
if $X_{set} = \varnothing$ **then**
    return $result$
end if
end while

The comparison of number of cutting off inequalities obtained from the convex hull of possible paths and greedy algorithm according to block ciphers Present [6], Klein [11] and Prince [7] is given below:

Table 2.2: The comparison of number of cutting off inequalities obtained from the convex hull of possible paths and greedy algorithm for S-boxes of ciphers Present, Klein and Prince

S-box	Greedy Algorithm Output Count	Convex Hull Inequality Count
Present	23	327
Klein	24	311
Prince	28	300
Prince Inversed	26	300

2.3.6 Reducing the Number of Cutting off Inequalities Using Mixed Integer Linear Programming

In 2017, Sasaki et al. suggested a new method that uses mixed integer linear programming to reduce the convex hull inequalities.

To convert this reduction problem to a mixed integer linear programming problem, first let the convex hull of possible paths of an S-box be $H_{set} = \{z_0, \ldots, z_n\}$ where any $z_i \in H_{set}$ represents a convex hull inequality. z_i is a binary variable that takes 1 if it is included in the model, or it takes 0 otherwise. As our aim here is to find the minimum number of convex hull inequalities to be included, our objective function is

$$\textbf{minimize } z_0 + z_1 + \cdots + z_{n-1} + z_n$$

To construct the constraints to minimize the objective function, one needs to find the eliminating inequalities for every impossible path. To make it more clear, let $imp_0, imp_1, \ldots, imp_k$ be impossible paths. Assume that their eliminating inequalities are $\{z_0, z_5, z_n\}$, $\{z_0, z_3, z_{n-1}\}, \ldots, \{z_3, z_{11}, z_{27}\}$, respectively. The table below shows which inequalities are active for their relating impossible path.

Table 2.3: Impossible paths with their eliminating convex hull inequalities

	z_0	z_1	z_2	z_3	z_4	z_5	$\ldots$	z_{11}	$\ldots$	z_{27}	$\ldots$	z_{n-1}	z_n
imp_0	1	0	0	0	0	1	$\ldots$	0	$\ldots$	0	$\ldots$	0	1
imp_1	1	0	0	1	0	0	$\ldots$	0	$\ldots$	0	$\ldots$	1	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\ldots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$
imp_k	0	0	0	1	0	0	$\ldots$	1	$\ldots$	1	$\ldots$	0	0

Then, the constraints from $imp_0, imp_1, \ldots, imp_k$ can be constructed as:

$$\begin{gathered} imp_0\text{: } z_0 + z_5 + z_n \geq 1 \\ imp_1\text{: } z_0 + z_3 + z_{n-1} \geq 1 \\ \vdots \\ imp_k\text{: } z_3 + z_{11} + z_{27} \geq 1 \end{gathered}$$

The sum of LHS must be greater than 1 to eliminate an impossible path by at least one convex hull inequality.

When modeled in this way, the convex hull inequality reduction problem may be solved more legitimately, instead of the biased perspective one in the greedy algorithm.

The comparison of the results from convex hull, greedy algorithm and reduction with MILP for ciphers Present, Klein, and Prince is given below:

Table 2.4: Comparison of number of cutting off inequalities collected from Convex Hull, Greed Algorithm and Reduction with MILP for S-boxes of ciphers Present, Klein and Prince

S-box	Convex Hull Inequality	Greedy Algorithm	Reduction with MILP
Present	327	23	21
Klein	311	24	21
Prince	300	28	22
Prince Inversed	300	26	22

In 2020, Boura et al. suggested a new algorithm for reducing the convex hull inequalities [8]. The algorithm aims to find new candidate inequalities using the linear combinations of convex hull inequalities, which needs to eliminate a new set of impossible paths other than the convex hull inequalities'. The algorithm takes possible paths of an S-box, and a positive integer k, which is the number of inequalities going to be combined to create a new candidate inequality. After taking the input, firstly, the convex hull inequalities H_{set} of its possible paths are calculated, and then the inequalities are assigned to an output set O_{set}. Then, for every possible path pos of S-box, and for every $\{C_1, \ldots, C_k\} \in P(H_{set}, k)$ such that any $C_i \in \{C_1, \ldots, C_k\}$ satisfied by pos with zero left hand side, C_{new} is computed from the vector addition of these inequalities, i.e $C_{new} = C_1 + \cdots + C_k$. Lastly, it checks that the set of impossible paths that the new candidate inequality C_{new} eliminates does not belong to the set of elimination sets of previous inequalities eliminate. If not, then the candidate is added to the output set O_{set}. The pseudocode of this algorithm is given below:

Algorithm 3 Extended cutting off inequality set of a given S-box

Input: the possible paths of a given S-box S_{pos}, positive integer k

Output: new cutting off inequalities O_{set} extended from convex hull of possible paths of a given S-box

$H_{set} \leftarrow ConvexHull(pos_{set})$

$O_{set} \leftarrow H_{set}$

for i **from** 0 **to** $len(pos_{set})$ **do**

  for $\forall\{C_1, \ldots, C_k\} \in C(H_{set}, k)$ **do**

    if pos_i satisfies every inequality in $\{C_1, \ldots, C_k\}$with zero LHS **then**

      $C_{new} \leftarrow C_1 + \cdots + C_k$

      if $E(C_{new}) \notin E(H_{set})$ **then**

        $O_{set} \leftarrow O_{set} \cup C_{new}$

      end if

    end if

  end for

end for

return O_{set}

The reason why Boura et al. did not check for every combination of $\{C_1, \ldots, C_k\} \in C(H_{set}, k)$ is, the combinations of the inequalities that do not intersect on a possible path of S-box do not eliminate a new set of impossible paths.

After finding a new set of cutting of inequalities from the extension of the convex hull of possible paths of given S-box using this algorithm, using this set, a new MILP model needs to be constructed to reduce to eventually find the smallest inequality set that eliminates the whole impossible paths just as in sasaki et al.'s work [18].

The comparison of the results from the convex hull, greedy algorithm, and reduction with MILP modeling for ciphers Present, Klein, and Prince is given below:

Table 2.5: Comparison of Inequality counts collected from Convex Hull, Greed Algorithm and Reduction with MILP Modelling for S-boxes of ciphers Present, Klein and Prince

S-box	C.H. of Possibles	Greedy Algorithm	Reduction with MILP	Extended Red. with MILP
Present	327	23	21	17
Klein	311	24	21	19
Prince	300	28	22	19
Prince Inv	300	26	22	19

CHAPTER 3

EXPERIMENTS TO REDUCE THE NUMBER OF CUTTING OFF INEQUALITIES

In this chapter, we will cover our experiments on SPN based block ciphers Present[6], Prince[7] and Klein[11]. After giving a brief description of those block ciphers, we will explain finding cutting off inequalities with our own idea, along with finding them with the techniques mentioned in the previous chapter.

3.1 Experiments to Reduce Number of Cutting Off Inequalities

We will now examine the techniques explained in the previous chapter regarding computing the number of cutting off inequalities along with our perspective. We have tried to develop the idea behind both Sasaki et al.'s and Boura et al.'s work, reducing the number of cutting off inequalities. We have primarily focused on the extension of the convex hull inequality set in Boura et al.'s work to achieve that. In their work, it is found that adding new inequalities to the convex hull set from taking the vector sum of some specific k combination of the inequalities in that, and then making a reduction via mixed integer linear programming ends up with less number of cutting off inequalities than the result of reduction from the original convex hull set for the ciphers referred above. Our work focused on finding a different extension set from the convex hull set to find a different reduced set, preferably less number of cutting off inequalities. We have considered two ideas about it. In the first one, multiplying the k combinations of inequalities instead of summing. Secondly, generalize the sum of combination in terms of changing the coefficients of the summed inequalities. In

Boura et al.'s paper, those coefficients were always 1. We have changed the coefficients in this method to obtain a different extended set. We could not come up with a better cutting off inequality set in terms of the inequality count in both methods.

In another approach, we have tried to combine the results of Moura et al.'s and Sun et al.'s work. In this scenario, we tried to focus on reducing the impossible paths to be eliminated, in conclusion, to achieve less cutting off inequality count.

3.1.1 Experiments on Present

Present is a 31 rounds SPN cipher designed by Bogdanov et al. [6]. It has 80-bit and 128-bit key variants. It has 64-bit block size, and its S-box size is 4-bit.

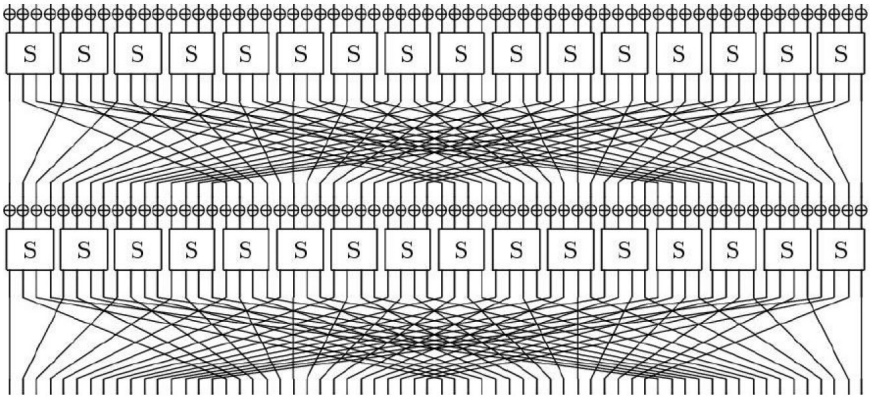

Figure 3.1: Two rounds encryption of Present

An encryption round of Present consists of addRoundKey, substitution, and permutation layers. In the first one, the 64-bit block state is simply XORed with the 64-bit round key. In substitution, the state is separated into 16 parts, and every part leads to a $4x4$ S-box. Lastly, in the permutation layer, the block is simply bitwise permuted into a new order.

Description of Present's S-box

Table 3.1: Present's S-box

X	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
S(X)	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

From its S-box, we first created its difference distribution table which can be found

below:

Table 3.2: Present S-box's Difference Distribution Table

Δ	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	***F***
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
2	0	2	0	2	0	0	0	4	0	2	2	0	0	0	2	2
3	0	2	0	2	0	0	4	0	0	2	2	0	0	0	2	2
4	0	0	0	0	0	0	0	0	0	0	4	4	2	2	2	2
5	0	0	0	0	2	2	2	2	0	0	4	4	0	0	0	0
6	0	2	0	2	0	4	0	0	0	2	2	0	2	2	0	0
7	0	2	0	2	4	0	0	0	0	2	2	0	2	2	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	4	4	4	4
9	0	4	4	0	0	0	0	0	0	4	0	4	0	0	0	0
A	0	0	2	2	2	0	0	2	4	0	0	0	0	2	0	2
B	0	0	2	2	0	2	2	0	4	0	0	0	2	0	2	0
C	0	4	4	0	2	2	2	2	0	0	0	0	0	0	0	0
D	0	0	0	0	2	2	2	2	0	4	0	4	0	0	0	0
E	0	0	2	2	0	2	2	0	4	0	0	0	0	2	0	2
F	0	0	2	2	2	0	0	2	4	0	0	0	2	0	2	0

Present's difference distribution table helps us to find its impossible paths (input and output pairs with zero appearances, i.e $P(i, o) = 0$), and its possible paths ($P(i, o) > 0$).

Cutting off Inequality Reduction in Present

As can be seen from the Table 3.2, Present has 159 impossible paths, and 97 possible paths. Extracting the convex hull of its possible paths gives us 327 inequalities to eliminate all of its impossible paths. Extending the convex hull using Boura et al.'s extension method, [8] gives 2303 new inequalities. Then, using Sasaki et al.'s reduction method [18], We constructed a MILP model to reduce the system. To optimize the MILP problem, we used Gurobi Optimizer [1]. It can be added to python as a library, and so it can be worked in that environment. Optimizing the problem ends up with 19 cutting off inequalities to eliminate the whole impossible paths.

Modifying the extension method with our ideas did not change the number of cutting off inequalities after all. On the other hand, classifying the impossible paths gave us new perspectives in terms of the number of constraints. we classified the impossible paths as **zero input - non zero output impossible paths**, **non zero input - zero output impossible paths**, **zero impossible paths with hamming weight lower than differential branch number** and others. Obviously, there is no common inequality

to eliminate the impossible paths referred as "others". On the other hand, the three other classes can be eliminated from some constraints referred in sun et al.'s, and mouha et al.'s [17] works. In [20], Sun et al. suggested two new constraints to adjust the MILP modeling in respect of eliminating the non zero input - zero output, and zero input - non zero output difference scenarios for S-boxes. The related constraints are:

$$s \cdot (x_0 + \cdots + x_{s-1}) - (y_0 + \cdots + y_{s-1}) \geq 0,$$
$$s \cdot (y_0 + \cdots + y_{s-1}) - (x_0 + \cdots + x_{s-1}) \geq 0.$$

Taking advantage of this idea, we omitted these 30 impossible paths from the previous 159 impossible paths, and so there remain 129 impossible paths to be eliminated. Secondly, we can benefit from Moura et al.'s idea of eliminating the impossible paths whose hamming weight is below S-box's differential branch number. The constraints concerning this operation are given below:

$$x_0 + x_1 + \cdots + x_{s-2} + x_{s-1} + y_0 + y_1 + \cdots + y_{s-2} + y_{s-1} \geq \mathbb{B} \cdot d,$$
$$d \geq x_0,$$
$$\vdots$$
$$d \geq x_{s-1},$$
$$d \geq y_0,$$
$$\vdots$$
$$d \geq y_{s-1},$$

Present's S-box's differential branch number is 3 and omitting the impossible paths whose hamming weight is less than 3, the remaining number of impossible paths is 118. Extending the convex hull inequalities according to these remaining impossible paths results in 1012 candidate inequalities. Reducing these inequalities gives us 15 cutting off inequalities along with Sun et al.'s, and Moura et al.'s constraints.

3.1.2 Experiments on Prince

Prince [7] is a 12 rounds SPN block cipher designed by Borghoff et al. in 2012. It is referred to as a low latency cipher and has 64-bit block state and 128-bit key

size. It has a symmetric structure such that it does not have extra implementation for its decryption, giving the key with α difference to its encryption method does the decryption, which is why the cipher is called $\alpha - symmetric$. A round of Prince consists of S-box layer, and a linear layer.

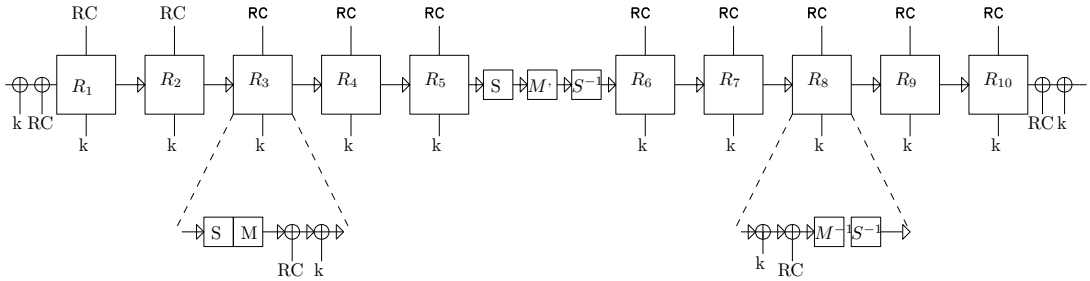

Figure 3.2: Internal Structure of Prince

S-box of Prince

Prince has a single $4x4$ S-box to substitute its state. However, starting from the seventh round, it uses inverse S-box. Therefore, the inverse S-box must also be examined.

Table 3.3: Prince's S-box

X	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
S(X)	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4

Table 3.4: Prince S-box's Difference Distribution Table

Δ	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	***F***
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	4	0	0	2	0	2	0	4	2	0	2	0	0	0	0
2	0	2	0	4	0	0	0	2	2	0	0	0	0	4	2	0
3	0	0	0	0	0	2	2	0	2	2	2	2	2	0	0	2
4	0	2	2	4	2	2	0	0	2	0	2	0	0	0	0	0
5	0	0	2	2	0	2	0	2	0	2	0	2	2	2	0	0
6	0	0	2	2	0	2	2	0	0	2	0	2	0	0	4	0
7	0	0	2	0	0	0	2	0	2	0	4	0	0	2	2	2
8	0	0	2	0	4	2	0	0	2	2	0	2	0	2	0	0
9	0	0	2	2	0	0	0	0	0	2	2	0	4	2	0	2
A	0	0	0	2	2	4	0	4	2	0	0	0	0	0	0	2
B	0	2	0	0	4	0	0	2	0	0	0	2	2	0	2	2
C	0	4	0	0	0	2	2	0	0	0	2	2	2	0	2	0
D	0	2	0	0	0	0	0	2	0	4	2	0	0	2	2	2
E	0	0	2	0	0	0	4	2	0	0	0	2	2	2	0	2
F	0	0	2	0	2	0	2	2	0	0	2	0	2	0	2	2

Table 3.5: Prince's Inverse S-box

X	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
S(X)	B	7	3	2	F	D	8	9	A	6	4	0	5	E	C	1

Table 3.6: Prince Inverse S-box's Difference Distribution Table

Δ	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	***F***
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	4	2	0	2	0	0	0	0	0	0	2	4	2	0	0
2	0	0	0	0	2	2	2	2	2	2	0	0	0	0	2	2
3	0	0	4	0	4	2	2	0	0	2	2	0	0	0	0	0
4	0	2	0	0	2	0	0	0	4	0	2	4	0	0	0	2
5	0	0	0	2	2	2	2	0	2	0	4	0	2	0	0	0
6	0	2	0	2	0	0	2	2	0	0	0	0	2	0	4	2
7	0	0	2	0	0	2	0	0	0	0	4	2	0	2	2	2
8	0	4	2	2	2	0	0	2	2	0	2	0	0	0	0	0
9	0	2	0	2	0	2	2	0	2	2	0	0	0	4	0	0
A	0	0	0	2	2	0	0	4	0	2	0	0	2	2	0	2
B	0	2	0	2	0	2	2	0	2	0	0	2	2	0	2	0
C	0	0	0	2	0	2	0	0	0	4	0	2	2	0	2	2
D	0	0	4	0	0	2	0	2	2	2	0	0	0	2	2	0
E	0	0	2	0	0	0	4	2	0	0	0	2	2	2	0	2
F	0	0	0	2	0	0	0	2	0	2	2	2	0	2	2	2

Prince's difference distribution table suggests that there are 106 possible paths and 150 impossible paths. Although it is not identical with its inverse, the inverse S-box of Prince, the number of possible and impossible paths are the same.

Cutting off Inequality Reduction in Prince

The convex hull of possible paths consists of 300 inequalities. Extending these inequalities results with 1042 new candidate inequalities, and reducing them via MILP gives us 19 cutting off inequalities. There are 120 impossible paths after omitting the nonzero-zero conflicting impossible paths and impossible paths whose hamming weight is lower than S-boxes differential branch number. Reducing the candidates with respect to these impossibles returns 18 cutting off inequalities with the constraints regarding zero - non zero input/output. Differential branch number constraints are not needed to be added as the related impossible paths are already eliminated by zero - non zero input/output constraints.

As mentioned earlier, the S-box of Prince has the same amount of possibles and impossibles with its inverse. Unsurprisingly, the numbers of convex hull inequalities are also the same. Extending and then reducing the inequalities results in 19 cutting off

inequalities. As impossibles with less weight than S-box's differential branch number are eliminated by zero - non zero input/output constraints, there are 120 impossible paths to be eliminated again. Following the same procedure gives us 18 cutting off inequalities.

3.1.3 Experiments on Klein

Klein [11] is a 12 rounds SPN block cipher designed by Gong et al. in 2011. It has 64 bits block size, and has 64, 80, and 128 bits keys with 12, 16, and 20 rounds, respectively. It divides its block into nibbles to do the operations, and these operations consist of subNibbles, rotateNibbles, and mixNibbles.

S-box of Klein

Table 3.7: Klein's S-box

X	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
S(X)	7	4	A	9	1	F	B	0	C	3	2	6	8	E	D	5

Table 3.8: Klein S-box's Difference Distribution Table

Δ	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	***F***
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	4	2	0	2	0	2	0	0	2	0	0	2	2
2	0	0	0	0	0	4	0	0	0	0	2	2	0	4	2	2
3	0	4	0	2	2	0	0	0	0	0	2	0	0	2	4	0
4	0	2	0	2	2	0	2	0	0	2	0	2	0	2	0	2
5	0	0	4	0	0	2	0	2	2	0	2	4	0	0	0	0
6	0	2	0	0	2	0	4	0	2	0	2	2	2	0	0	0
7	0	0	0	0	0	2	0	2	2	2	0	0	2	0	4	2
8	0	2	0	0	0	2	2	2	2	2	0	2	0	0	0	2
9	0	0	0	0	2	0	0	2	2	0	0	2	2	2	2	2
A	0	0	2	2	0	2	2	0	0	0	4	0	2	0	2	0
B	0	2	2	0	2	4	2	0	2	2	0	0	0	0	0	0
C	0	0	0	0	0	0	2	2	0	2	2	0	4	2	0	2
D	0	0	4	2	2	0	0	0	0	2	0	0	2	2	0	2
E	0	2	2	4	0	0	0	4	0	2	2	0	0	0	0	0
F	0	2	2	0	2	0	0	2	2	2	0	0	2	2	0	0

From its DDT, one can extract its number of impossible and possible paths 150 and 106, respectively.

Cutting off Inequality Reduction in Klein

The number of convex hull inequalities of its possible paths is 311, and extending and then reducing these inequalities gives us 19 cutting off inequalities. Since its differential branch number is 2, like in Prince, omitting just zero - nonzero input/output differences from its impossible path set and reducing the inequalities according to it gives us 18 cutting off inequalities.

Table 3.9: Performance Comparison of Models constructed with Boura et al.'s Extension Method and Our adjustments

			Time (s.)	
Cipher	**Rounds**	**# of active S-Boxes**	**With Boura et al.'s**	**With our Adjustments**
Present	10	20	4423	752
Prince	5	19	153	129
Klein	4	15	682	597

CHAPTER 4

CONCLUSION

Block ciphers are popular encryption methods today. With its increasing popularity, new methods of cryptanalysis continue to emerge. Therefore, it is very important to calculate the safety margins for both the analyzer and the attacker. The main and most important attacks on block ciphers are differential and linear cryptanalysis attacks. The most important parameter that these attacks depend on is the number of active S-boxes. Therefore, calculating the minimum number of active S-boxes for a given number of rounds is one of the crucial safety parameters for a block cipher.

The minimum active S-box calculation can be done mathematically, or programmatically. In this thesis, automatically calculating the minimum active S-box with mixed integer linear programming (MILP) is investigated.

Chapter 2 investigates the works done by Mouha et al. [17], Sun et al. [20],[21],[22], Sasaki et al. [18], and Boura et al. [8]. It is explained how MILP modeling can be done by utilizing difference propagation, structural properties of encryption operations (S-boxes specifically).

Chapter 3 covers our contribution to reduce the number of constraints to model an S-box operation. We categorized impossible input-output difference pairs of an S-box according to their certain properties, and then obtained a new set of constraints other than the related works mentioned in chapter 2 to eliminate them. Then the MILP models from recent techniques and our adjustments are compared for the block ciphers Present [6], Prince [7], and Klein [11]. A slightly improvement in performance is observed for the models of these ciphers. These results are given in Table 3.9.

To optimize this problem with MILP, the difference propagation within the cipher must be modeled correctly. Therefore, The studies on this subject in the literature are examined and it is emphasized how the security mechanisms of block ciphers can be modeled to a MILP problem. We have focused on a certain problem in this subject, which is to find less number of inequalities to omit the impossible paths of an S-box in a cipher. To develop a new idea, we inspired some previously works done related to this subject. We categorized the impossible paths of an S-box to find a divide and conquer approach for the MILP model of S-box. Using this approach, we had able to achieve a better performance to find minimum number of active S-boxes for the ciphers Present, Prince, and Klein. The results are given in Table 3.9.

4.1 Discussions on Future Work

We will continue our work to improve the performance of finding minimum number of active S-boxes using MILP. Our work was to finding a cutting off inequality set of impossible paths of an S-box with less size. There are also other studies to improve the perfomrance made by Ilter et al. in [27]. Their work is on finding new XOR modelings in order to increase the overall performance of automatically calculating security bounds of ciphers Prince and Klein. Therefore we aim to examine the modelings of other operations of the ciphers in order to increase the overall performance.

On the other hand, Lightweight cryptography standardization work of NIST has been going on for some time. The remaining ciphers at the end of the second round of the competition were determined recently [23]. Calculating the security parameters of these ciphers with MILP is also among our future studies.

Finally, apart from SPN-based block ciphers, calculating the security parameters of ARX (addition-rotation-XOR) based ciphers with MILP is one of the studies we want to study on. The main difficulty in examining ARX based ciphers with MILP is that modular sum in large fields is difficult to follow with MILP because it contains so many possible outcomes and therefore their model would simply create too many inequalities.

REFERENCES

[1] Gurobi optimizer, the fastest solver, https://www.gurobi.com/, Aug 2021.

[2] E. Andreeva, B. Bilgin, A. Bogdanov, A. Luykx, F. Mendel, B. Mennink, N. Mouha, Q. Wang, and K. Yasuda, Primates v1, Submission to CAESAR, 2014.

[3] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, Camellia: A 128-bit block cipher suitable for multiple platforms—design andanalysis, in *International workshop on selected areas in cryptography*, pp. 39–56, Springer, 2000.

[4] E. Biham and A. Shamir, Differential cryptanalysis of des-like cryptosystems, Journal of CRYPTOLOGY, 4(1), pp. 3–72, 1991.

[5] B. Bilgin, A. Bogdanov, M. Knežević, F. Mendel, and Q. Wang, Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware, in *International Conference on Cryptographic Hardware and Embedded Systems*, pp. 142–158, Springer, 2013.

[6] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe, Present: An ultra-lightweight block cipher, in *International workshop on cryptographic hardware and embedded systems*, pp. 450–466, Springer, 2007.

[7] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, et al., Prince–a low-latency block cipher for pervasive computing applications, in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 208–225, Springer, 2012.

[8] C. Boura and D. Coggia, Efficient milp modelings for sboxes and linear layers of spn ciphers, IACR Transactions on Symmetric Cryptology, 2020(3), pp. 327–361, Sep. 2020.

[9] J. Daemen and C. Clapp, Fast hashing and stream encryption with panama, in *International Workshop on Fast Software Encryption*, pp. 60–74, Springer, 1998.

[10] J. Daemen and V. Rijmen, The wide trail strategy, in *The Design of Rijndael*, pp. 123–147, Springer, 2002.

[11] Z. Gong, S. Nikova, and Y. W. Law, Klein: a new family of lightweight block ciphers, in *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, pp. 1–18, Springer, 2011.

[12] J. Jean, I. Nikolic, and T. Peyrin, Kiasu v1, Submitted to the CAESAR competition, 2014.

[13] J. Jean, I. Nikolić, and T. Peyrin, Joltik v1. 3, CAESAR Round, 2, 2015.

[14] J. Jean, I. Nikolic, T. Peyrin, and Y. Seurin, Deoxys v1. 41, Submitted to CAESAR, 124, 2016.

[15] E. B. Kavun, M. M. Lauridsen, G. Leander, C. Rechberger, P. Schwabe, and T. Yalçın, Prøst v1, CAESAR Round, 1, 2014.

[16] M. Matsui, Linear cryptanalysis method for des cipher, in *Workshop on the Theory and Application of of Cryptographic Techniques*, pp. 386–397, Springer, 1993.

[17] N. Mouha, Q. Wang, D. Gu, and B. Preneel, Differential and linear cryptanalysis using mixed-integer linear programming, in *International Conference on Information Security and Cryptology*, pp. 57–76, Springer, 2011.

[18] Y. Sasaki and Y. Todo, New algorithm for modeling s-box in milp based differential and division trail search, in P. Farshim and E. Simion, editors, *Innovative Security Solutions for Information Technology and Communications*, pp. 150–165, Springer International Publishing, Cham, 2017, ISBN 978-3-319-69284-5.

[19] Y. Sasaki, Y. Todo, K. Aoki, Y. Naito, T. Sugawara, Y. Murakami, M. Matsui, and S. Hirose, Minalpher v1, CAESAR Round, 1, 2014.

[20] S. Sun, L. Hu, L. Song, Y. Xie, and P. Wang, Automatic security evaluation of block ciphers with s-bp structures against related-key differential attacks, in *International Conference on Information Security and Cryptology*, pp. 39–51, Springer, 2013.

[21] S. Sun, L. Hu, M. Wang, P. Wang, K. Qiao, X. Ma, D. Shi, L. Song, and K. 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, Cryptol. ePrint Archive, Report, 747, pp. 2014–45, 2014.

[22] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, Automatic security evaluation and (related-key) differential characteristic search: application to simon, present, lblock, des (l) and other bit-oriented block ciphers, in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 158–178, Springer, 2014.

[23] M. Sönmez Turan, K. McKay, D. Chang, Çalık, L. Bassham, J. Kang, and J. Kelsey, Status report on the second round of the nist lightweight cryptography standardization process, Jul 2021.

[24] C. Tezcan, Improbable differential attacks on present using undisturbed bits, Journal of Computational and applied mathematics, 259, pp. 503–511, 2014.

[25] D. Watanabe, T. Owada, K. Okamoto, Y. Igarashi, and T. Kaneko, Update on enocoro stream cipher, in *2010 International Symposium On Information Theory & Its Applications*, pp. 778–783, IEEE, 2010.

[26] S. Wu, H. Wu, T. Huang, M. Wang, and W. Wu, Leaked-state-forgery attack against the authenticated encryption algorithm ale, in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 377–404, Springer, 2013.

[27] M. İlter. and A. Selcuk., A new milp model for matrix multiplications with applications to klein and prince, in *Proceedings of the 18th International Conference on Security and Cryptography - SECRYPT,*, pp. 420–427, INSTICC, SciTePress, 2021, ISBN 978-989-758-524-1, ISSN 2184-7711.