



**Brno University of  
Technology**



**University of L'Aquila**

---

**Double-Degree Master's Programme - InterMaths  
Applied and Interdisciplinary Mathematics**

**Master of Science  
Mathematical Engineering**

Brno University of Technology (BUT)

**Master of Science  
Mathematical Engineering**

University of L'Aquila (UAQ)

**Master's Thesis**

*Type-preserving Matrices and Block Cipher Security*

**Supervisor**

Dr. Riccardo Aragona

*Riccardo Aragona*

---

**Candidate**

Tunmbi OKEDIRAN

*Tunmbi Okedirani*

---

Student ID (UAQ): 274451

Student ID (BUT): 243842

**Academic Year 2021/2022**



# BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

## FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

## INSTITUTE OF MATHEMATICS

ÚSTAV MATEMATIKY

# TYPE-PRESERVING MATRICES AND BLOCK CIPHER SECURITY

MATICE ZACHOVÁVAJÍCÍ TYP A BEZPEČNOST BLOKOVÝCH ŠIFER

## MASTER'S THESIS

DIPLOMOVÁ PRÁCE

## AUTHOR

AUTOR PRÁCE

Tunmbi Olayemi Okediran

## SUPERVISOR

VEDOUCÍ PRÁCE

Dr. Riccardo Aragona

BRNO 2022

BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Mechanical Engineering

MASTER'S THESIS

Brno, 2022

Tunmbi Olayemi Okediran



# Assignment Master's Thesis

Institut: Institute of Mathematics  
Student: **Tunmbi Olayemi Okediran**  
Degree programm: Mathematical Engineering  
Branch:  
Supervisor: **Dr. Riccardo Aragona**  
Academic year: 2021/22

As provided for by the Act No. 111/98 Coll. on higher education institutions and the BUT Study and Examination Regulations, the director of the Institute hereby assigns the following topic of Master's Thesis:

## Type-preserving Matrices and Block Cipher Security

### Brief Description:

Recently, a new property for the linear layer of a block cipher is introduced for guaranteeing protection against some algebraic attacks. Such layers satisfying this property are called non-type-preserving. An interesting result is to characterize such layers by providing a list of necessary and sufficient conditions on the structure of their underlying binary matrices. It is worth pointing out that several families of linear maps are non-type-preserving, including the mixing layers of AES, GOST and PRESENT. Finally, it is possible to prove that the group generated by the round functions of an SPN cipher with addition modulo  $2^n$  as key mixing function is primitive if its mixing layer satisfies this property.

### Master's Thesis goals:

After a description of both the mathematical and cryptographic background, the known results regarding type preserving matrices and their connections with symmetric cryptography will be presented.

Eventually, we will try to extend such results to the case of actions of the key which is different from the sum modulo  $2^n$  which is already established.

### Recommended bibliography:

ARAGONA, R., MENEGHETTI, A. Type-preserving matrices and security of block ciphers, *Advances in Mathematics of Communications* 13(2), 235-251(2019).

ARAGONA, R., CALDERINI, M., TORTORA, A., TOTA, M. Primitivity of PRESENT and other lightweight ciphers, *Journal of Algebra and Its Applications* 17(6), 1850115 (2018), [16 pages].

ARAGONA, R., CARANTI, A., DALLA VOLTA, F., SALA, M. On the group generated by the round functions of translation based ciphers over arbitrary finite fields, *Finite Fields and Their Applications* 25, pp. 293-305 (2014).

CARANTI, A., DALLA VOLTA, F., SALA, M. On some block ciphers and imprimitive groups, *Applicable Algebra in Engineering, Communication and Computing* 20, pp. 339-350 (2009).

Deadline for submission Master's Thesis is given by the Schedule of the Academic year 2021/22

In Brno,

L. S.

---

prof. RNDr. Josef Šlapal, CSc.  
Director of the Institute

---

doc. Ing. Jaroslav Katolický, Ph.D.  
FME dean

## Abstrakt

Zavedli jsme novou vlastnost směšovacích vrstev blokových šifer. Tato vlastnost se nazývá non-type-preserving a zaručuje odolnost vůči algebraickým útokům na základě imprimitivity skupiny generované kruhovými funkcemi.

Uvažovali jsme binární matici odpovídající směšovací vrstvě a dali jsme potřebné a dostatečné podmínky pro binární matici, která zajišťuje typ nezachovávající vlastnost. Pak jsme ukázali, že některé reálné šifry splňují tyto nezbytné a postačující podmínky, a proto jsou typ nezachovávající. Uvažované šifry byly GOST, PRESENT a AES.

Nakonec jsme v kapitole 4 ukázali, že pokud míchací vrstva šifry SPN, která používá adiční modulo  $2^n$  pro míchání klíčů, nezachovává typ, pak je skupina generovaná funkcí round primitivní.

## Summary

We introduced a new property of the mixing layers of block ciphers. This property is called non-type-preserving and it guarantees resistance to algebraic attacks based on imprimitivity of the group generated by the round functions.

We considered the binary matrix corresponding to the mixing layer and gave necessary and sufficient conditions on the binary matrix that ensures non-type-preserving property. Then we showed that some real-life ciphers satisfy those necessary and sufficient conditions and so are non-type-preserving. The ciphers considered were GOST, PRESENT, and AES.

Lastly, in chapter 4 we showed that if the mixing layer of SPN cipher that uses addition modulo  $2^n$  for key mixing is non-type-preserving then the group generated by the round function is primitive.

## Klíčová slova

### Keywords

Block cipher, Block cipher security, Mixing layer, Imprimitivity attack, Type-preserving matrix.

OKEDIRAN, T. O. *Matice zachovávající typ a zabezpečení blokových šifer*. Brno: Vysoké učení technické v Brně, Faculty of Mechanical Engineering, 2022. 46 s. Dr. Riccaro Aragona, Ph.D.





I declare that I have written this diploma thesis Type-preserving Matrices and Block Cipher Security on my own, under the direction of my supervisor, Dr. Riccardo Aragona, Ph.D., and using the sources listed in the bibliography.

Tunmbi Olayemi OKEDIRAN



Blessed be the LORD my strength, who teaches my hands to war, and my fingers to fight. I want to say a big thank you to my supervisor, Dr. Riccardo Aragona for serving as a vista through which I peeped into the world of groups and cryptography. I would also like to thank him for the opportunity to learn from him.

I express my profound gratitude to the all InterMaths staffers and all the distinguished lecturers that taught me. Also to my colleagues turned friends, Joshua Adeleke, Michael Asante, Lanre Moshood, Garcia Sebastian and Samson Quaye I say a big thank you for your companionship during my stay here in Brno.

I would like to say a big thank you to my mom, sister, brothers, Mr. Ojo and Pst. Niyi Adebayo for their support. I say a big thank you to my girlfriend for the patient endurance she exhibited during the course of my master's program. Lastly, I bid all African students studying abroad who have suffered any form of prejudice to remember that *per ardua ad astra*.

Tunmbi Olayemi OKEDIRAN

# Contents

<b>1</b>	<b>Introduction</b>	<b>14</b>
1.1	History of Cryptography . . . . .	14
1.2	Classification of Cryptographic Algorithms . . . . .	15
1.3	Literature Review . . . . .	16
1.4	Goal of Thesis . . . . .	17
1.5	Thesis Outline . . . . .	17
<b>2</b>	<b>Block Ciphers</b>	<b>19</b>
2.1	Why block ciphers? . . . . .	19
2.2	Design principles and Classification of Block Ciphers . . . . .	19
2.2.1	Fiestel Networks (FN) . . . . .	20
2.2.2	Substitution Permutation Network (SPN) . . . . .	21
2.2.3	Round Functions . . . . .	25
2.3	Cryptanalysis . . . . .	25
2.3.1	Attack Models . . . . .	26
2.3.2	Cryptanalysis Techniques . . . . .	26
2.4	Summary . . . . .	27
<b>3</b>	<b>Preliminary results and Type-preserving matrices</b>	<b>28</b>
3.1	Round functions and Primitivity . . . . .	30
3.2	Type-preserving Matrices . . . . .	34
3.2.1	Examples of Non-type-preserving mixing layer . . . . .	38
<b>4</b>	<b>Applications</b>	<b>41</b>
4.1	PRIMITIVITY OF AN SPNMOD CIPHER . . . . .	41
4.2	PRIMITIVITY OF GOST-LIKE CIPHERS . . . . .	42
<b>5</b>	<b>Conclusion</b>	<b>44</b>

# List of Figures

2.1	DES Feistel block cipher . . . . .	20
2.2	GOST block cipher . . . . .	21
2.3	A typical four rounds of an SPN . . . . .	22

# 1. Introduction

This chapter introduces the scope of the thesis which is block cipher security, in particular we will define a particular type of matrices called type-preserving matrices related to block cipher security. In Section 1.1 we discuss the history of cryptography. In Section 1.2 we discuss the classification of cryptography algorithms and we review of former literature in Section 1.3. Section 1.4 state the goal of the thesis and lastly we give the outline of the thesis in Section 1.5.

## 1.1. History of Cryptography

Hieroglyph was used in ancient Egypt, as a formal writing system of the scribes. Hieroglyph was used to communicate messages on behalf of the king, only the scribes are privy to the meaning of the message. The codes are pictorial, the meaning of this pictorial symbols are known only by the scribes. Similarly the Yoruba people of West Africa, send confidential messages from one king to another using symbols, symbols which only the priest understood. This system of communication is called Àrokò. These two systems of communication strive for confidentiality in communication. This in essence is what cryptography provides. Cryptography is simply the science of writing and sending secret messages over an open channel. Open channel in that anybody can have access to this message but only the intended receiver should understand the message.

We can see that cryptography is an ancient art and it is also of utmost importance. Over the years the science of sending encrypted messages have metamorphosed into complex and sophisticated algorithms. The message that is sent confidentially is called plaintext and the encrypted message is called ciphertext. Although hieroglyph might be 4000 years old, in the 100 BC the roman emperor Julius Caesar was credited with the invention of an encryption method, this is known as Caesar cipher. It was used when conveying secret messages to roman army generals during a war. A cipher is an algorithm used for encryption and decryption. Caesar cipher is an example of substitution ciphers, in a substitution cipher each alphabetic character of the plaintext is changed to another character, another example of a substitution cipher is the affine cipher.

Caesar ciphers shifts each alphabet by 3, for instance the English alphabets A - Z are labelled 0 - 25 respectively. The encryption function is  $\epsilon = (x + n) \bmod 26$  where  $n \in \mathbb{N}$  and  $x$  the label of a character of the plaintext. The decrypting process is carried out using the function  $d = (y - n) \bmod 26$ ,  $y$  is the label of a character of the ciphertext. The simplicity of this cipher has both advantage and disadvantage. Advantage because it's easy to implement but the disadvantage arise from the fact that it has 26 key choices which means it is prone to exhaustive key search.

The affine cipher is also an example of substitution ciphers, affine cipher is an upgrade on the Caesar cipher, the relative sophistication of the affine cipher provides more security. The encryption function is,  $\epsilon = (ax+b) \bmod 26$ , where  $a, b \in \mathbb{N}$  and  $\gcd(a, 26) = 1$ ,  $x$  is the label of the character in the plaintext. The decryption function is,  $d = (a^*y - a^*b) \bmod 26$ , where  $a^*$  is the multiplicative inverse of  $a \bmod 26$ . The condition  $\gcd(a, 26) = 1$  assures

that the encryption function is injective, in other words the existence of the inverse of  $a \bmod 26$ . The key choices in an affine cipher are 312.

It is easy to see that the security of both Caesar cipher and affine cipher depend on the secrecy of the system and not on the encryption key. This is because once the system is known, then encrypted messages can easily be decrypted. In fact, substitution ciphers can be broken by using the frequency of letters in the language. Vigenere in the 16th century also designed a cipher which was the first cipher using an encryption key. In this cipher, the encryption key was repeated multiple times spanning the entire message, and then the ciphertext was produced by adding the message character with the key character modulo 26 [1]. Vigenere cipher encrypts blocks of plaintext unlike the shift ciphers. In the Vigenere cipher the secrecy of the message depends on the secrecy of the encryption key, rather than the secrecy of the system.

During the end of first World War, a German engineer named Arthur Scherbius invented the enigma machine. Enigma is a mechanical encryption device which used about four rotors. Each rotor had twenty-six possible initial setting, the key was the initial setting of the rotors. The German forces used the Enigma for confidential communication during the second World War. However a group of three polish cryptographers succeeded in breaking the Enigma, and these three cryptographers sent the technique to the British. Alan Turing worked extensively on the Enigma machine and devised a means of obtaining a daily key.

Anciently cryptography was mainly used for military purpose but after the second World War, the commercial world saw the usefulness of cryptography, that it can be used to hide information from competitors. In the twenty-first century, cryptographer is widely used in our day to day activities, including sending of emails, cash withdrawal from the ATM, secure web browsing, blockchain and so on.

## 1.2. Classification of Cryptographic Algorithms

There are many ways of classifying cryptographic algorithms, in this thesis we will employ the classification based on the key used for encryption. In this class of classification we have symmetric encryption and asymmetric encryption.

Asymmetric encryption uses a public key for data encryption and a private key for decrypting the ciphertext. The plaintext is encrypted using the public key of the recipient, the recipient after receiving the ciphertext, then he/she can decrypt it using his/her own private key. Examples include the Diffie-Hellman exchange method, Elliptical Curve Cryptography and Rivest Shamir Adleman (RSA) algorithm. Asymmetric encryption are mainly used for key exchange because they are not efficient due to slow speed and as a result of this slow speed, the network is overburdened during the encryption and decryption process.

Symmetric encryption uses a the same key for both encrypting the plaintext and decrypting the ciphertext. This key is however a private key, i.e. it is known only by the genuine

### 1.3. LITERATURE REVIEW

entities involved in the communication. Examples of this type of encryption are broadly stream ciphers and block ciphers. The private key is shared with recipient using any of the key exchange protocol, such as Diffie-Hellman exchange method, elliptic curve cryptography and RSA algorithm. The main focus of this thesis is on block cipher which is a type of symmetric encryption.

### 1.3. Literature Review

Since the introduction of the first civilian block cipher (LUCIFER) in the 1970s by Horst Fiestel was a response to the announcement made by the National Bureau of Standards (NBS), which is now called the National Institute of Standard and Technology (NIST), the design by Horst Fiestel was presented by IBM as a proposal. The algorithm proposed by IBM in 1974 was adopted as the Data Encryption Standard (DES) and like earlier stated it was based on Fiestel's Lucifer cipher [22].

Loopholes were found in the design of DES not because it was badly designed but due to rigorous study of DES which led to revolutionary ideas in the cryptanalysis of these types of cryptographic algorithms. Biham and Shamir's [6] differential cryptanalysis in the 1980s and Matsui's linear cryptanalysis in the early 90s were arguably chiefs amongst these ideas. These two classes of cryptanalysis and advances in cryptanalysis till recent times has rendered DES rather insecure in many applications.

Due to the insecurity that DES posed, NIST announced in 1997 the replacement of DES. After a five-year standardization processing the Rijndael cipher [13] - named after two Belgian cryptographers, Joan Daemen and Vincent Rijmen, who also turned in a proposal in response to this announcement - was selected for the Advanced Encryption Standard (AES) in 2001 amongst fifteen competing designs. Since its selection in 2001 and despite the rigorous analysis on this cryptosystem, it is still considered secure.

DES and AES fall under the Block cipher class this is because the algorithm is designed to work on grouped bits of a fixed length called blocks. Most modern block ciphers are obtained by composition of round functions and are either Fiestel Networks (FN) or Substitution Permutation Networks (SPN). In a round function, there are three operations that occur. One of these operations is key mixing - this is combines the message with the corresponding round key - which is mostly done by XOR.

The major drawback of XOR is the linearity, which makes the cipher more susceptible to differential linear cryptanalysis. This could be prevented by using addition modulo  $2^n$  where  $n$  is an integer, which intuitively could seem like a good practice. In [18] it is showed that this intuition is quiet valid after constructing two SPN block ciphers one with the traditional key mixing of XOR and the other with the key mixing of additional modulo  $2^n$ , which was named GPig1 and Gpig2 respectively. The author checked from an experimental point of view that the bias of the linear cryptanalysis dropped exponentially fast, which translate to an increase in non-linearity. Another cipher, which is not SPN



but FN, using addition modulo  $2^n$  as key mixing is GOST [14].

In [17] it was showed that some weaknesses of block cipher are inherent in the group-theoretical properties. For example if DES were closed under functional composition then strengthening DES through using multiple encryptions would be equivalent to single encryption and if the corresponding group is small then there exists a vulnerability due to birthday-paradox attacks. Investigation of the possibility of the existence of trapdoors in ciphers whose encryption functions generate an imprimitive group or an affine group was done by Paterson in [19] and by Calderini in [9] respectively and they showed that these properties can be turned into an efficient attack. This inspired a branch of research in symmetric cryptography focused on showing the primitivity of group generated by encryption functions.

## 1.4. Goal of Thesis

We aim to ascertain that round functions using additional modulo  $2^n$  as key mixing function is watertight against algebraic attacks based on imprimitivity of the group that is generated by this type of round functions.

We introduced a mixing layer property called non-type-preserving. We will show that this property guarantees resistance to imprimitivity attacks.

## 1.5. Thesis Outline

The organization of this thesis is as follows:

Chapter 2 introduces the design principles of block ciphers, the usefulness and importance of a block cipher. Here we discuss about different approaches to cryptanalysis of block ciphers which are bruteforce, differential cryptanalysis, linear cryptanalysis and algebraic cryptanalysis. We also give the classification of block ciphers based on their design principle. To close the chapter we summarized the whole chapter.

Chapter 3 presents some general definitions and known results which served as the foundation of the algebraic cryptanalysis done in this thesis. We also discussed on the primitivity of the group generated by the round function, we introduce the notion of black boxes, ruled boxes and white boxes used later in the thesis. We also define what we mean by type-preserving matrices, and gave the necessary and sufficient condition that guarantees this property. We give examples of real-life ciphers that possess non-type-preserving mixing layers.

Chapter 4 presents how the type-preserving property of known ciphers coupled with other assumptions on the cipher guarantee primitivity of the whole cipher.

## 1.5. *THESIS OUTLINE*

Chapter 5 is the conclusion and also outline of possible future research directions.

## 2. Block Ciphers

The design principles are explained after general usefulness and classification of block ciphers, this is simply to ease the passage to the algebraic analysis. Then lastly we will discuss the cryptanalysis of block ciphers.

Even though this section is about the general introduction and design of block ciphers, the understanding of this chapter will help in the algebraic properties of round functions.

### 2.1. Why block ciphers?

Cryptography studies communications in the presence of an adversary, which translates to high need for secrecy. This is what cryptosystems strive to provide. A confidential channel of communication – where messages are only readable by the genuine recipient and also the sender and for any third party it will be a meaningless piece of information–, one of the cryptographic primitives used to achieve this goal is the block cipher.

The algorithm used in the design of block cipher works on grouped bits as mentioned earlier, this is the reason for the name block cipher. One of the many advantages of block ciphers is that if one letter of the plaintext is changed, this change affects the whole ciphertext.

Algorithms used in cryptography are divided into different classes depending on the key. Block ciphers are symmetric key ciphers, this type of algorithm use the same key for encrypting and decrypting. This key has to be shared between the sender and genuine recipient in a secured way. Symmetric key ciphers are logically used for authentication and confidentiality.

The challenge of symmetric key ciphers (in particular block ciphers) would have been the secret way of sharing the key but this is well handled by the public-key algorithms, which are cleverly designed in a way that allows parties to share with each other the public part of a key, to construct a shared private key. We continue this introduction by discussing in a formal way the design principles of block ciphers.

### 2.2. Design principles and Classification of Block Ciphers

One of the highly desirable property that block ciphers possess is confusion and diffusion, this was brought to bear by Claude Shannon [20]. Diffusion means that a single change in the plaintext should affect several characters of the ciphertext and vice-versa, it's basically to make the plaintext bits and key bits relation difficulty to analyse by any attacker. Confusion makes sure that each character of the ciphertext is dependent on several parts of the key. It is concerned with the relationship between ciphertext and key. In order to fully explain the working principles of block ciphers, we will discuss the two main algorithms used for block ciphers,

## 2.2. DESIGN PRINCIPLES AND CLASSIFICATION OF BLOCK CIPHERS

### 2.2.1. Feistel Networks (FN)

Feistel Networks (FN) named after the German-born physicist and cryptographer Horst Feistel, is a symmetric key algorithm used in the design of DES and LUCIFER. The plaintext and the key are the main input. FN is an iterative procedure that is repeated for a required number of times. The number of rounds is dependent on the bits of the plaintext. The iterative procedure consists of a round function, this is responsible for the encryption process with the aid of subkeys – subkeys are gotten from the master key using a certain function called key-schedule – .

The plaintext is divided into two halves  $L_0$  and  $R_0$ , this is the initial halves. For each round  $i = 0, 1, 2, \dots, r$  we compute

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= L_i \oplus F(R_i, K_i) \end{aligned} \quad (2.1)$$

where  $\oplus$  is the XOR and the ciphertext is  $(R_{n+1}, L_{n+1})$ ,  $F(\cdot, \cdot)$  is the round function,  $R_i$  is the right half of the  $i$ th round and  $L_i$  is the left side of the  $i$ th round and  $K_i$  the subkey derived by the action of the key schedule on the master key. Example of a block ciphers that uses the design of FN are DES and GOST cipher, we give the design principle of GOST cipher after the diagram.

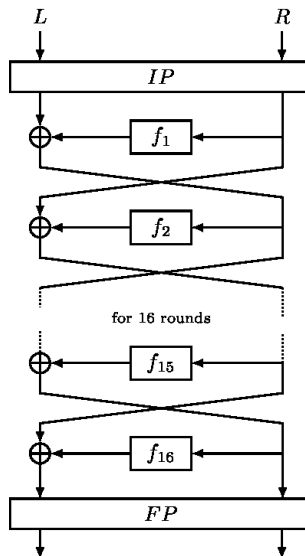


Figure 2.4: DES Feistel block cipher

### GOST cipher

GOST[11] cipher is an algorithm that is designed after the FN structure, it encrypts 64-bit plaintexts into 64-bit ciphertexts with the aid of a key,  $K = (K_1, \dots, K_8)$  such that  $K_i$  is 32-bit, for  $i = 1, \dots, 8$ . These  $K_i$  are called partial keys and are used sequentially in each of the 32 rounds. GOST was designed by the Soviet Union (now Russia) in 1970s, the design allows for the secrecy of both the key and the 8 S-boxes.

The 64-bit plaintext is divided into two halves  $L_0$  and  $R_0$  of size 32-bit each. The key mixing operation is carried out on  $R_0$ , i.e

$$R_0 + K_1 \pmod{2^{32}} = R_*$$

$R_*$  has 32 bits and it is passed into the 8 S-boxes, the resulting 32-bit string is then passed into the permutation layer which does 11 bits right rotation, the resulting 32-bit string, say  $R$ , is then XORed with  $L_0$ , this gives  $R_1$ , and  $R_0$  will now be  $L_1$ . More in general the  $(i - 1)th$  round is

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus F(R_{i-1}, L_{i-1}, K_j). \end{aligned}$$

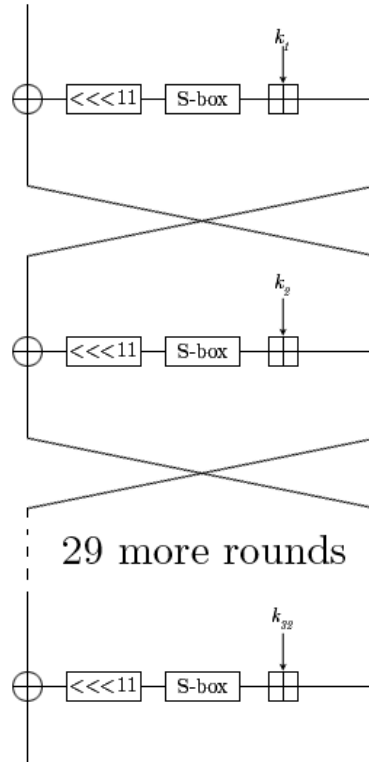


Figure 2.2: GOST block cipher

for  $i \in \{1, \dots, 32\}$ ,  $j \in \{1, \dots, 8\}$ . The same procedure is repeated for the 32 rounds. The partial keys  $K_i$  are used in the order

$$K_1, \dots, K_8, K_1, \dots, K_8, K_1, \dots, K_8, K_1, \dots, K_8, K_8, \dots, K_1$$

for each rounds. Both  $L_{32}$  and  $R_{32}$  are concatenated to form the final output which is the 32-bit ciphertext.

In the next subsection we discuss the other structure with which block ciphers are designed with, this is the Substitution Permutation Network (SPN).

### 2.2.2. Substitution Permutation Network (SPN)

The plaintext is divided into parts called bricks after key mixing – key mixing is simply the plaintext being XORed with the subkey – each bricks is passed into a substitution box (S-box) achieving confusion. The bricks are put together again and the corresponding block is passed into a permutation later achieving diffusion. This procedure is carried out repeatedly for a number of rounds with the subkey being changed for each round.

## 2.2. DESIGN PRINCIPLES AND CLASSIFICATION OF BLOCK CIPHERS

The most studied application of SPN is the Rijndael block cipher that won the AES competition announced by NIST. This block cipher algorithm was developed by Vincent Rijmen and Joan Daemen[13]. We discuss briefly on a typical structure of substitution boxes and permutation layer and give the design principle of AES.

- **substitution boxes (S-boxes):** This is the layer that is responsible for confusion. It can be considered as a function from a vector space  $\mathbb{F}_2^m$  to  $\mathbb{F}_2^n$  i.e  $S : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ , where  $n$  and  $m$  can be the same or different but mostly not greater than a byte ( 8 bits ). This function is constructed to be invertible for being able to decrypt. S-boxes are have some algebraic properties like bijection, non-linearity, completeness and high algebraic degree but just for mentioning some of them. They are constructed using different methods
- **Permutation boxes (P-boxes) :** This linear transformation is applied on the output of the S-boxes. The main purpose of the permutation boxes is that diffusion be achieved. Diffusion hides any forms of statistical relationship between the ciphertext and plaintext. According to Shannon [20] diffusion refers to dissipating the statistical structure of plaintext over the bulk of ciphertext, so that bit redundancy apparent in the plaintext won't be apparent in the ciphertext.

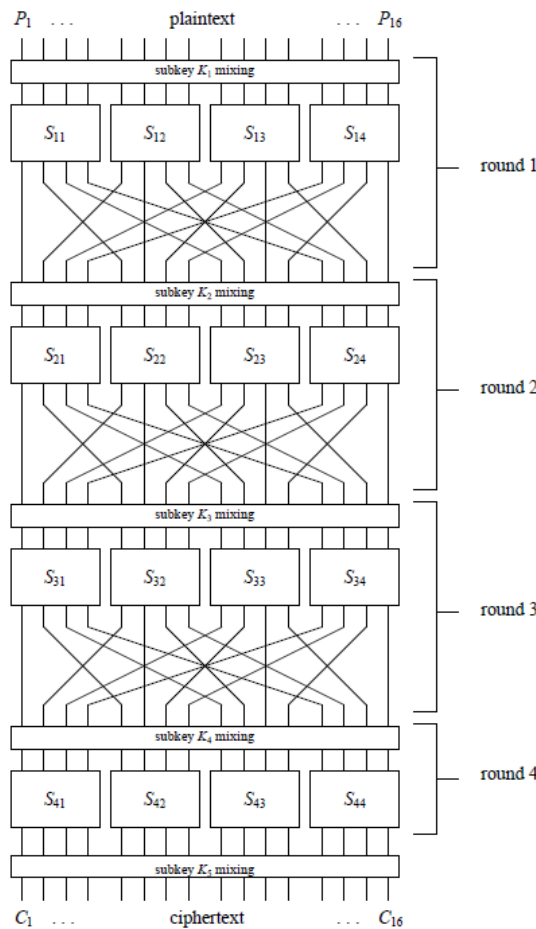


Figure 2.3: A typical four rounds of an SPN

## Advanced Encryption Standard(AES)

We give a brief explanation of the working principle of the AES. In the year 2000, the winner of AES was announced to be the algorithm submitted by Daemen and Rijmen, this algorithm was called Rijndael [13]. The AES uses repeated numbers of rounds to obtain security, this approach is similar to the one used in DES, however the design of AES is not the same as that of DES. Each round in AES consists of substitution, permutation and key mixing, this is traditionally done with XOR. The encryption and decryption operation of AES are distinct, also worthy of note is that the mathematical structure behind the design of AES is based on arithmetic in the field  $\mathbb{F}_{2^8}$  (i.e  $GF(2^8)$ ). The arithmetic carried out on  $GF(2^8)$  is performed using polynomial arithmetic modulo the irreducible polynomial  $X^8 + X^4 + X^3 + X + 1$ . This polynomial was the choice of Rijndael.

AES is highly desirable not just because of a better security it offers but also because it works not only with longer plaintext, it has different sizes namely, 128 bits, 192 bits and 256 bits. We will however consider only the 128 bits in the explanation of the working principle we will give here. The explanation of the working principle was taken from [21].

Rijndael operates on a byte matrix called the state matrix. The state matrix  $S$ , is a  $4 \times 4$  matrix of bytes

$$S = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix},$$

note that each  $s_{i,j}$ ,  $i, j = 0, \dots, 3$  is 8-bits. The round key is also held in a  $4 \times 4$  matrix

$$K_i = \begin{bmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix}.$$

The operation of the Rijndael is divided into four steps. We will give the steps for encryption and also give a pseudo-code of the encryption function and decryption function.

1. **SubBytes** : This is the first step, it involve the non-linear layer which is the S-box. The S-box of Rijndael has a simple structure which assures resistance to differential cryptanalysis and it also gives an assurance that there is no existence of any trapdoor. Mathematically, the operation of the S-box is first to take the multiplicative inverse of each byte of the state matrix  $s_{i,j}$ ,  $i, j = 0, \dots, 3$ . However, since there's no multiplicative inverse for 0 byte, the convention is to map it to zero.

## 2.2. DESIGN PRINCIPLES AND CLASSIFICATION OF BLOCK CIPHERS

This multiplicative inverse we will represent by  $x = [x_7, \dots, x_0]$ . Secondly is to perform an affine  $\mathbb{F}_2$  transformation given below;

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

The new byte is  $y$ . For the decryption process, we take the inverse of the affine transformation and  $y$  is used in place of  $x$ .

2. **ShiftRows**: The main essence of this step is to cause the columns of the state matrix to interact with each other when the process is repeated. The ShiftRows performs a cyclic shift both while encrypting and decrypting, the operation of the ShiftRows during encryption is given below;

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,1} & s_{1,2} & s_{1,3} & s_{1,0} \\ s_{2,2} & s_{2,3} & s_{2,0} & s_{2,1} \\ s_{3,3} & s_{3,0} & s_{3,1} & s_{3,2} \end{bmatrix}.$$

The operation is in the opposite direction when decrypting.

3. **MixColumns** : The MixColumns makes sure that each row in the state matrix interacts with each other over a number of rounds. Both the ShiftRows and the MixColumns ensure that every byte interact one with another, so that a little change to the input state will affect the output state greatly.
4. **AddRoundKey** : The roundkey addition is simply XORing the state matrix with the round key matrix in a bitwise manner.

We give the pseudo-code for encryption and decryption below.

Rijndael encryption	Rijndael decryption
AddRoundKey ( $S, K_0$ )	AddRoundKey ( $S, K_{10}$ )
<b>for</b> $i = 1$ <b>to</b> 9 <b>do</b>	InverseShiftRows ( $S$ )
SubBytes( $S$ )	InverseSubBytes ( $S$ )
ShiftRows( $S$ )	<b>for</b> $i = 9$ <b>downto</b> 1 <b>do</b>
MixColumns( $S$ )	AddRoundKey( $S, K_i$ )
AddRoundKey( $S, K_i$ )	InverseMixColumns( $S$ )
<b>end</b>	InverseShiftRows ( $S$ )
SubBytes ( $S$ )	InverseSubBytes ( $S$ )
ShiftRows ( $S$ )	<b>end</b>
AddRoundKey ( $S, K_{10}$ )	AddRoundKey ( $S, K_0$ )



### 2.2.3. Round Functions

Mathematically, block ciphers can be defined as a family of key-dependent permutation of the plaintext space  $V$  i.e  $\{\varepsilon_k \mid \varepsilon_k : V \rightarrow V, k \in \mathcal{K}\} \subseteq \text{Sym}(V)$ , where  $|V| \leq |\mathcal{K}|$ . The encryption function  $\varepsilon_k$  is the composition of  $r$  round function  $\varepsilon_{k_h}$  induced by the subkey  $k_h$ . Each subkey is generated by a public procedure  $\varphi : \{1, 2, \dots, r\} \times \mathcal{K} \rightarrow V$ , such that  $\varphi(h, k) = k_h$  is the  $h$ -th round key, given the key  $k$ . The round function  $\varepsilon_k$  is defined thus

$$\varepsilon_{k_h} = \gamma \lambda \sigma_{k_h} \quad (2.2)$$

- $\gamma : V \rightarrow V$  is a non-linear permutation, called parallel S boxes which acts in parallel on each bricks  $V_j$ , i.e  $(x_1, x_2, \dots, x_n)\gamma = ((x_1, \dots, x_m)\gamma_1, \dots, (x_1, \dots, x_m)\gamma_\delta)$ , where  $V = V_1 \times \dots \times V_\delta$  and  $V_j = \mathbb{F}_2^m$  for each  $j$ .
- $\lambda \in \text{Sym}(V)$  is a linear map called mixing layer that is permutation layer.
- $\sigma_{k_h} : V \rightarrow V$  is the key mixing function, that is a combination of the plaintext and the corresponding subkey.

From now on, when there is no risk of getting confused, we will delete the subscript  $h$  and write  $\varepsilon_k$  instead of  $\varepsilon_{k_h}$ .

## 2.3. Cryptanalysis

The cryptanalyst sole purpose is to exploit the weaknesses of a cryptosystem in order to break the security provided by the cryptographic primitive. This is done by mathematical algorithms, known as attack, directed against cryptosystems which at best could get the secret key. In order to get this secret key the cryptanalyst studies the cryptosystem, first to know how it works and at the end for finding a technique to weaken the it. The cryptanalysis introduced here is directed against block ciphers. Even though the main goal of the cryptanalyst is to get the secret key of the cryptosystem this is often an unattainable problem. There is a classification of the attacks based on the information retrieved, enumerated below in an increasing order of strength according to Alkhzaimi in[2];

- **Distinguishing algorithm:** This algorithm simply states the type of design used in a cryptosystem. For instance, if we have two black boxes which one consists of a block cipher implementation and the other just randomly chosen permutation on  $\mathbb{F}_2^n$ . A distinguishing algorithm strives to effectively determine one that contains a block cipher.
- **Local deduction:** The cryptanalyst here can generate the plaintext from the ciphertext and vice versa
- **Global deduction:** Without any knowledge about the key, the cryptanalyst will be able to develop an equivalent encryption function and decryption function that yield the same result as the encryption and decryption function used in the cryptosystem.
- **Total break:** This is the most desired jewel of any cryptanalyst where the most desirable information of a cryptosystem – the secret key – is obtained by the cryptanalyst.

## 2.3. CRYPTANALYSIS

### 2.3.1. Attack Models

Attack models are the level of knowledge or accessibility to information that an attacker is privy to when trying to carry out an attack on a cryptographic primitive. These models are chosen based on the success probability they pose for a specific cryptanalysis method and also on the complexity of the attack. In [2] different classes of attack models were as listed below;

- **Ciphertext Only Attack (COA)** : This is the most realistic of the classes. It assumes that the only information an attacker possesses is about the ciphertext and by analysing the ciphertext the attacker might be able to come up with a partial or complete information about the plaintext, in example of this class of attack model is the bruteforce method.
- **Chosen Ciphertext Attack (CCA)** : This model also makes use of plaintext knowledge but this is only after decrypting some selected ciphertext. A cryptanalysis method that makes use of this model is the differential cryptanalysis.
- **Chosen Plaintext Attack (CPA)** : This model also makes use of plaintext knowledge but this is only after encrypting some selected plaintext. A cryptanalysis method that makes use of this model is the differential cryptanalysis.
- **Related Key Attack** : Both the knowledge of the ciphertext and the plaintext is made available to the attacker in this model. And these ciphertexts are under different unknown but related keys that possess a chosen (or unknown) relation with the key to be recovered.

### 2.3.2. Cryptanalysis Techniques

There exist some techniques for cryptanalysing block cipher, one of the best known method is differential cryptanalysis introduced by Biham and Shamir[6]. Linear cryptanalysis is another technique widely studied and it was introduced by Matsui at the EUROCRYPT '93. Also there is a recent approach called the algebraic cryptanalysis. Below we give an overview of such attack method;

- **Bruteforce** : This is also known as exhaustive key search, it is a general technique for all cryptographic algorithms, here the cryptanalyst tries all possible keys in order to decrypt the ciphertext. On an average the cryptanalyst will try  $2^{n-1}$  different keys before breaking a cipher that has a key size of  $n$  bits but on a bad day the trials can be as many as  $2^n$ . The computational complexity of an attack is denoted by  $O(2^n)$ . For a cipher to be assumed to be secure, the best known attack against it must have at least the same computational capability as the bruteforce[8].
- **Differential cryptanalysis** : This technique can also be used against any cryptographic algorithm that is designed with a non-changing round function in an iterative procedure. Differential cryptanalysis was the first known attack that could recover, albeit theoretically, the DES key in a time lesser than the exhaustive key search computational time[6]. It exploits – if there exist any – the high probability of certain occurrences of plaintext differences and differences of ciphertext[16]. Let  $\bar{X}$  and  $\tilde{X}$  be two distinct plaintexts with the same number of bits such that

$\bar{X} = [\bar{x}_1, \dots, \bar{x}_n]$  and  $\tilde{X} = [\tilde{x}_1, \dots, \tilde{x}_n]$  and let  $\bar{Y}$  and  $\tilde{Y}$  be their respective ciphertext such that  $\bar{Y} = [\bar{y}_1, \dots, \bar{y}_n]$  and  $\tilde{Y} = [\tilde{y}_1, \dots, \tilde{y}_n]$ , then the plaintext difference is  $\Delta X = \bar{X} \oplus \tilde{X}$  and the ciphertext difference is  $\Delta Y = \bar{Y} \oplus \tilde{Y}$  where  $\oplus$  is XOR.

- **Linear cryptanalysis** : It is a known plaintext attack, however with the cryptanalyst having no way of selecting the plaintext and corresponding ciphertext. Linear cryptanalysis exploits any high probability occurrences of linear expressions involving plaintexts, ciphertexts and subkeys[16]. It was however unsuccessful in rendering DES unsecured, since it requires  $2^{47}$  known plaintexts before getting the key.
- **Algebraic cryptanalysis** : This technique is relatively new, it explores the algebraic properties of the encryption functions and the cipher is converted to a system of equations. The properties of the group generated by encryption functions are analysed for inherent trapdoors see for example[6], also the cipher can be converted into a system of polynomial equations over Galois field  $\mathbb{F}_2$ , sometimes over other rings and this polynomial equations can be solved, according to Bard in[5]. In this thesis we will focus on exploiting the trapdoor from some group properties.

## 2.4. Summary

Here we discussed a rather general introduction to block ciphers by discussing the design principles and the classification which it falls under and the usefulness of block ciphers in cryptography. While also we talked about the cryptanalysis of block ciphers.

We discussed in detail the working principles and the components with the underlining mathematical methods employed in this design which will come in handy when exploring their algebraic properties.

### 3. Preliminary results and Type-preserving matrices

In this chapter we discuss some known results that are vital for the imprimitivity of group. This property is used in the attack that will be discussed in the next chapter. Moreover, we will give the definition of group generated by the round function of a block cipher and we present some properties, finally we define the so-called type-preserving matrices [4]. We start this chapter with the definition of vector spaces, group and some specific groups. Lastly we discussed some results already established in [10] and [3].

**Definition 3.1.** A Vector space  $V$  (also called linear space) over a scalar field  $K$  is a set whose elements are called vectors, on which two operations are defined namely vector addition (+) and scalar multiplication ( $\cdot$ );

- Vector addition:  $V \times V \ni (\mathbf{u}, \mathbf{v}) \rightarrow \mathbf{u} + \mathbf{v} \in V$
- Scalar multiplication:  $K \times V \ni (\lambda, \mathbf{v}) \rightarrow \lambda \cdot \mathbf{v}$

$\lambda \in K, \forall \mathbf{u}, \mathbf{v} \in V$ , and the following properties are satisfied;

1.  $\forall \mathbf{u}, \mathbf{v}, \mathbf{w} \in V$ ,
  - $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$ ,
  - $\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$
2.  $\forall \mathbf{v} \in V, \exists 0 \in V$  such that  $0 + \mathbf{v} = \mathbf{v} + 0 = \mathbf{v}$ ,
3.  $\forall \mathbf{v} \in V, \exists -\mathbf{v} \in V$  such that  $\mathbf{v} + (-\mathbf{v}) = (-\mathbf{v}) + \mathbf{v} = 0$ ,
4.  $\forall \mathbf{u}, \mathbf{v} \in V$  and  $\lambda, \mu \in K$ ,
  - $1 \cdot \mathbf{v} = \mathbf{v}$
  - $(\lambda + \mu) \cdot \mathbf{v} = \lambda \cdot \mathbf{v} + \mu \cdot \mathbf{v}$
  - $\lambda \cdot (\mathbf{u} + \mathbf{v}) = \lambda \cdot \mathbf{u} + \lambda \cdot \mathbf{v}$ .

Vector spaces form a group under the operation of vector addition (+). The plaintext space  $V = \{0, 1\}^n$ , where  $n \in \mathbb{N}$  and key space  $\mathcal{K} = \{0, 1\}^p$ , where  $p \in \mathbb{N}$  are vector spaces.

**Definition 3.2.** A pair  $(G, *)$ , where  $G$  is a non-empty set and  $* : G \times G \rightarrow G$  is a binary operation, is called a group if the following conditions are satisfied;

- Associative property:  $\forall a, b \in G, (a * b) * c = a * (b * c)$
- Identity element:  $\forall a \in G, \exists e \in G$  such that,  $a * e = e * a = a$  ( $e$  is called the identity element.)
- Existence of inverse: every element  $a \in G$  has exactly one inverse, i.e  $\forall a \in G, \exists ! b \in G$  such that  $a * b = b * a = e$

### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

A group  $G$  such that  $|G| < \infty$  is called a finite group. A pair  $(H, *)$ , where  $H \subset G$ , that satisfies the three conditions listed above, is called subgroup of  $G$ . If the operation  $*$  is also commutative i.e  $a * b = b * a$ , for any in the underlining set, then the group is called Abelian group.

**Definition 3.3.** The set of  $n \times n$  invertible matrices, together with the operation of ordinary multiplication is called general linear group of order  $n$ , denoted by  $GL_n$ .  $GL_n$  forms a group under this operation since the product of two invertible matrices is again invertible, an invertible matrix possesses an inverse, with identity matrix as the identity element of the group.

**Definition 3.4.** The symmetric group on a set  $V$  denoted by  $Sym(V)$  is the set of all the permutations of  $V$ . The pair  $(Sym(V), \circ)$ , where  $\circ$  is function composition, is a group. Finally a group  $G$  is called permutation group, if  $G$  is a subgroup of  $Sym(V)$  for some set  $V$ .

**Definition 3.5.** Let  $V$  be a set and  $G$  be a group, any surjective homomorphism  $\alpha : G \rightarrow Sym(V)$ , is called action of  $G$  on  $V$ . for each  $g \in G$  and  $v \in V$ , the action of  $g$  on  $v$  is denoted by  $vg$ .

**Definition 3.6.** Let  $G$  be a group acting on  $V$ , the orbit of  $v \in V$  is defined as,  $vG \stackrel{\text{def}}{=} \{gv \mid g \in G\}$ . Every action of a group on a set decomposes the set into orbits.

**Definition 3.7.** The stabilizer of  $v \in V$ , under the action of group  $G$ , is the set of all  $g \in G$  which give a fix permutation of  $v$  i.e.  $G_v = \{g \in G \mid vg = v\}$ .

**Definition 3.8.** The group  $G$  is transitive on the set  $V$ , if  $V$  is non-empty and there is exactly one orbit.

**Proposition 3.1.** For  $V \neq \emptyset$ , an action of  $G$  on  $V$  is transitive if and only if, given  $u, v \in V$ ,  $\exists g \in G$  such that  $v = ug$  [12].

*Proof:*

Suppose the action is transitive, so there is one orbit. Given  $u \in V$ , its orbit must fill up  $V$ , so  $v = ug$ ,  $\forall v \in V$  and some  $g \in G$ . Conversely, suppose that  $\forall u, v \in V$  we can write  $v = ug$  for some  $g \in G$ . Fix  $u \in V$ . Since every  $v \in V$  has the form  $vg$  for some  $g \in G$ , every  $v$  is in the orbit of  $u$ . Thus  $V$  has only one orbit.  $\square$

The consequence of Proposition 3.1 above is an equivalent definition of transitivity group i.e. a group  $G$  is said to be transitive on  $V$  if for each  $v, u \in V$ ,  $\exists g \in G$  such that  $v = ug$ .

**Definition 3.9.** Let  $\mathcal{B} = \{U \mid U \subseteq V\}$  be a partition of  $V$  (i.e  $V$  is a disjoint union of the sets of  $\mathcal{B}$ ),  $\mathcal{B}$  is said to be  $G$ -invariant if for any  $B \in \mathcal{B}$  and  $g \in G$  we have  $Bg \in \mathcal{B}$ . A partition  $\mathcal{B}$  is called trivial if  $\mathcal{B} = V$  or  $\mathcal{B} = \{v \mid v \in V\}$ .

**Definition 3.10.** Any non-trivial,  $G$ -invariant partition  $\mathcal{B}$  of  $V$  is called block system for the action of the group  $G$ . Every  $B \in \mathcal{B}$  is called imprimitivity block, if  $G$  is transitive on  $V$  and there is no block system, then the action of group  $G$  is called primitive on  $V$ .

### 3.1. ROUND FUNCTIONS AND PRIMITIVITY

**Lemma 3.1.** *A block of imprimitivity is the orbit  $vH$  of a proper subgroup  $H < G$  that properly contains the stabilizer  $G_v$  for some  $v \in V$ .*

**Lemma 3.2.** *If  $T$  is a transitive subgroup of  $G$ , then a block system for  $G$  is also a block system for  $T$*

## 3.1. Round functions and Primitivity

The algebraic attack we are considering in this thesis is based on the imprimitivity of the permutation group that is generated by the block cipher, this type of attack was described in [19].

We can define a block cipher as

$$\mathcal{C} = \{\varepsilon_k : V \longrightarrow V \mid k \in \mathcal{K}\}, \quad (3.1)$$

where  $\mathcal{K}$  is the key space,  $V$  is the plaintext space, in particular  $V$  is a vector space over  $\mathbb{F}_2$  of dimension  $n$ , and  $\varepsilon_k$  is a permutation of  $V$  and  $\mathcal{C} \subseteq \text{Sym}(V)$ . We can define a permutation group when we consider the group generated by  $\mathcal{C}$ ,

$$\Gamma(\mathcal{C}) \stackrel{\text{def}}{=} \langle \varepsilon_k \mid k \in \mathcal{K} \rangle \leq \text{Sym}(V). \quad (3.2)$$

From definition we see that  $\Gamma(\mathcal{C})$  strongly depends on the key-schedule function which makes the study of  $\Gamma(\mathcal{C})$  really difficult, for this reason, the study of key-schedule function is outside the scope of this thesis. Paterson and Kaliski in ([19],[17]), showed that it is interesting to study  $\Gamma(\mathcal{C})$  in order to know weakness of a block cipher. Since  $\Gamma(\mathcal{C})$  is difficult to study we define another group containing  $\Gamma(\mathcal{C})$  that is  $\Gamma_\infty(\mathcal{C})$  where the roundkey varies in  $k \in \mathcal{K}$  and

$$\Gamma_\infty(\mathcal{C}) \stackrel{\text{def}}{=} \langle \varepsilon_{h,k} \mid k \in \mathcal{K} \rangle. \quad (3.3)$$

Notice that the round key vary for each round  $h = 1, \dots, r$  and so  $\Gamma_\infty(\mathcal{C})$  allows us to ignore the effect of the key schedule.

In this thesis we are interested on the ciphers called substitution permutation networks, where each round function is

$$\varepsilon_{h,k} = \gamma\lambda\sigma_{k_h}, \quad (3.4)$$

with  $\gamma$  a non-linear permutation,  $\lambda$  a linear permutation and  $\sigma_{k_h}$  the key mixing function using the subkey generated by the key schedule function for the  $h$ th-round. So,  $\Gamma_\infty(\mathcal{C})$  can be expressed in terms of these round function i.e

$$\Gamma_\infty(\mathcal{C}) = \langle \gamma\lambda\sigma_{k_h} \mid 1 \leq h \leq r, k \in \mathcal{K} \rangle, \quad (3.5)$$

sometimes we will denote  $\Gamma_\infty(\mathcal{C})$  by  $\Gamma_\infty$ . The set  $V$ , the space of plaintext, possesses two different group structures, namely the traditional addition modulo 2 (i.e XOR), denoted by  $\oplus$  and the addition modulo  $2^n$  which is denoted by  $\boxplus$ , which is used in the key mixing function. XOR makes  $V$  a vector space over  $\mathbb{F}_2$ . Now we define the additional

### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

modulo  $2^n$ . Let  $a = (a_0, a_1, \dots, a_n)$  and  $b = (b_0, b_1, \dots, b_n)$  be elements of  $V$ , i.e  $a_i, b_i \in \{0, 1\}$  for each  $i$ . Let us define  $\boxplus$ ,

$$\begin{aligned} a \boxplus b &= (a_0 + a_1 2 + \dots + a_{n-1} 2^{n-1}) + (b_0 + b_1 2 + \dots + b_{n-1} 2^{n-1}) \\ &\equiv c_0 + c_1 2 + c_2 2^2 + \dots + c_{n-1} 2^{n-1} \pmod{2^n} \\ &= (c_0, c_1, \dots, c_{n-1}), \end{aligned} \tag{3.6}$$

where  $c_i \in \{0, 1\}$ . It is easily to prove that the pair  $(V, \boxplus)$  is isomorphic to the group  $\mathbb{Z}_{2^n}$  of integer modulo  $2^n$ . We will denote this group by  $(\mathbb{Z}_{2^n}, \boxplus)$ . Below are some elementary fact we will repeatedly make use of, for more details and proofs of the following results see [3].

**Lemma 3.3.** *The subgroups of  $(\mathbb{Z}_{2^n}, \boxplus)$  are linearly ordered; they are  $\langle 2^q \rangle$ , for  $q \in [0, n]$ .*

We defined  $\mathcal{T}$  as the group of  $\boxplus$ -translations on  $V$ ,

$$\mathcal{T} \stackrel{\text{def}}{=} \{\sigma_k : v \rightarrow v \boxplus k \mid k \in V, \sigma_k : V \rightarrow V\}, \tag{3.7}$$

notice that  $\sigma_0(v) = v, \forall v \in V$ , this means that 0 is the orbit of every element in  $V$ , so this makes  $\mathcal{T}$  transitive on  $V$ .

**Lemma 3.4.**

$$\Gamma_\infty = \langle \mathcal{T}, \gamma\lambda \rangle. \tag{3.8}$$

$\Gamma_\infty$  is transitive on  $V$

Proof:

If we set  $k = 0$ , then  $\gamma\lambda\sigma_0 = \gamma\lambda$ , and  $\gamma\lambda \in \Gamma_\infty$ , so  $(\gamma\lambda)^{-1} \in \Gamma_\infty$ . Now for all  $k \in V$   $(\gamma\lambda)^{-1}\gamma\lambda\sigma_k = \sigma_k$  and  $\sigma_k \in \Gamma_\infty$ .  $\square$

Since the map  $v \rightarrow \sigma_v$  preserves the structures between  $(V, \boxplus)$  and  $\mathcal{T}$ , this fact leads to the following two well known results which are taken from [10].

**Lemma 3.5.** *The subgroups of  $\mathcal{T}$  are of the form*

$$\{\sigma_u : u \in U\},$$

where  $U$  is a subgroup of  $(V, \boxplus)$

**Lemma 3.6.** *If  $\Gamma_\infty$ , acting on  $V$ , has a block system, then this consists of the cosets of a  $\boxplus$ -subgroup of  $V$ , that is, it is of the form*




$$\{W \boxplus v : v \in V\},$$

where  $W$  is a non-trivial, proper subgroup of  $(V, \boxplus)$ .

### 3.1. ROUND FUNCTIONS AND PRIMITIVITY

Paterson in [19] gave an example of a pseudo DES cipher that is resistant to linear cryptanalysis and differential cryptanalysis but be easily broken by algebraic attack since the round functions of this pseudo DES cipher generate an imprimitive group, so it can be easily broken. This is because imprimitivity of the group can be used as means of constructing a hidden trapdoor. The study carried out in [19] aroused cryptanalysts' intrigue in studying imprimitivity of the groups generated by round functions.

Now we consider the results and definitions given in[3], we also adopt the same notation that was used therein. According to lem2q, a subgroup  $D$  of  $\mathbb{Z}_{2^n}$  is of the form  $\langle 2^q \rangle$ , for  $q \in [0, n[$ . Therefore we represent each of the elements  $d$  of  $D = \langle 2^q \rangle$  as an element of  $\mathbb{F}_n^2 = \mathbb{F}_2^q \times \mathbb{F}_2^{n-q}$  of the form  $0_{[0, q-1]} || d_{[q, n-1]} \in \mathbb{F}_2^{n-q}$ . We shall use the same compact notation used in[3]:

1. A white box  denotes a subset of  $\mathbb{F}_2^m$  of cardinality 1,
2. A blue box (will be called ruled box)  denotes a subset of  $\mathbb{F}_2^m$  of cardinality  $1 < t < 2^m$ ,
3. A black box  denotes the full set  $\mathbb{F}_2^m$ .

We will say that a box has white, ruled or black type.

**Definition 3.11.** Let  $D$  be a subset of

$$\mathbb{F}_n^2 = V_1 \times V_2 \times \cdots \times V_\delta,$$

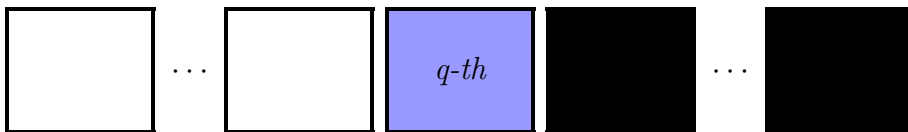
where each space  $V_i$  are subspaces of  $V$  and with dimension  $m$  (such that  $m \times \delta = n$ ) are called bricks. We shall say that  $D$  has a type if

$$D = (D \cap V_1) \times (D \cap V_2) \times \cdots \times (D \cap V_\delta).$$

This type will be a sequence of  $\delta$  white, ruled or black boxes, where the  $i - th$  block is the projection of  $D$  on  $V_i$  [3].

**Remark 3.1.** lem2q states that a subgroup  $D$  of  $\mathbb{Z}_{2^n}$  has the form  $\langle 2^q \rangle$ , for  $q \in [0, n[$ . Thus subgroup  $D = \langle 2^q \rangle$  has one of the following two types.

1. The first is when  $q \not\equiv 0 \pmod{m}$ , there is a ruled box which is the containing the  $q$ -th bit i.e



2. The second one is when  $q \equiv 0 \pmod{m}$ ;



### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

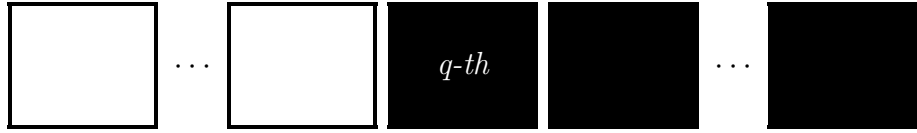
(a) when  $q = 0$ , here the subgroup is the full group  $\mathbb{Z}_{2^n}$  and all are black boxes i.e



(b) when  $q = n$ , here the subgroup is  $\{0\}$  and all are white boxes i.e



(c) when  $q \neq 0$  and  $q \neq n$ , here the subgroup has  $n_w$  white boxes and  $n_b = \delta - n_w$  black boxes, where  $n_w$  and  $n_b$  are integer such that  $q = n_w m$  and  $0 \leq n_w, n_b \leq \delta$  i.e



From Remark 3.1, dittoed [3], we set the bounds for the number of white, black and ruled boxes which are respectively  $n_w$ ,  $n_b$  and  $n_r$ . The type of any subgroup  $D$  of  $\mathbb{Z}_{2^n}$  can be associated with the number of each boxes i.e  $(n_w, n_r, n_b)$ . Subsequently we, with a abuse of notation, represent the type of  $D$  as  $(n_w, n_r, n_b)$ . Also from Remark 3.1 we can find information about the bounds of  $n_w$ ,  $n_b$  and  $n_r$ . We see that  $n_r$  can appear at most once from the first condition in Remark 3.1.  $n_b, n_w$  can appear at most  $\delta$  times which we can see from the 2(a) and 2(b) of 3.1 respectively, thus we have the bounds;

$$\begin{aligned} n_w + n_b + n_r &= \delta \\ 0 \leq n_w &\leq \delta \\ 0 \leq n_b &\leq \delta \\ 0 \leq n_r &\leq 1 \end{aligned} \tag{3.9}$$

The behaviour of the sum  $\boxplus$  with respect to the type is considered in the next lemma, this also is a result from [3].

**Lemma 3.7.** *If  $D$  is a subgroup of  $\mathbb{Z}_{2^n}$  and  $v \in \mathbb{Z}_{2^n}$  then  $D$  and  $v \boxplus D$  have the same type.*

Proof: Any element  $d$  of  $D = \langle 2^q \rangle$  has a binary representation of the form;

$$d = 0_{[0, q-1]} \parallel d_{[q, n-1]},$$

we can also divide  $v$  into two halves i.e  $v = v_{[0, q-1]} \parallel v_{[q, n-1]}$ , then  $\forall d \in D$

$$\begin{aligned} v \boxplus d &= (v_{[0, q-1]} \parallel v_{[q, n-1]}) \boxplus (0_{[0, q-1]} \parallel d_{[q, n-1]}) \\ &= v_{[0, q-1]} \boxplus 0_{[0, q-1]} \parallel v_{[q, n-1]} \boxplus d_{[q, n-1]} \\ &= v_{[0, q-1]} \parallel v_{[q, n-1]} \boxplus d_{[q, n-1]} \end{aligned}$$

$0_{[0, q-1]}$  is a zero vector of length  $q$  and  $d_{[q, n-1]}$  ranges in  $\mathbb{F}_2^{n-q}$ , so does  $v_{[q, n-1]} \boxplus d_{[q, n-1]}$ . Since  $d$  is arbitrary in  $D$  then  $D$  and  $v \boxplus D$  are of the same type.  $\square$

### 3.2. Type-preserving Matrices

The round function was earlier defined as  $\varepsilon_{h,k} = \lambda\gamma\sigma_k$  where  $\lambda$  is the mixing layer,  $\gamma$  is the permutation layer and  $\sigma_k$  is the subkey addition. We had earlier mentioned in Subsection [Round Functions](#) the main goal of the components of the round function and we said that the main goal of  $\lambda$  is the diffusion. In this section we will study the diffusion properties of an invertible mixing layer  $\lambda$ , the linear layer  $\lambda$  mixes the subspaces  $V_i$ ,  $\forall i = 1, \dots, \delta$  of  $V$ , which are also called bricks. Let us consider  $\Lambda$  the matrix of size  $n$  corresponding to  $\lambda$ , where  $n$  is the dimension of  $V$ . The dimension of the subspaces  $V_i$ ,  $\forall i = 1, \dots, \delta$  is  $m$  such that  $n = m\delta$ . The matrix  $\Lambda$  is given by;

$$\Lambda = \begin{bmatrix} \Lambda_{1,1} & \Lambda_{1,2} & \cdots & \Lambda_{1,\delta} \\ \Lambda_{2,1} & \Lambda_{2,2} & \cdots & \Lambda_{2,\delta} \\ \vdots & \vdots & \ddots & \vdots \\ \Lambda_{\delta,1} & \Lambda_{\delta,2} & \cdots & \Lambda_{\delta,\delta} \end{bmatrix}.$$

Notice that  $\Lambda$  is a block matrix of dimension  $\delta \times \delta$ , and each submatrix  $\Lambda_{i,j}$ ,  $1 \leq i, j \leq \delta$  of  $\Lambda$  is of dimension  $m \times m$ . If for any  $i \neq j$ ,  $\Lambda_{i,j} = 0$  then  $\lambda\gamma$  is a parallel map. For each submatrix, we shall use the notation;

$$\Lambda_{(i_1,j_1):(i_2,j_2)} \stackrel{\text{def}}{=} \begin{bmatrix} \Lambda_{i_1,j_1} & \cdots & \Lambda_{i_1,j_2} \\ \vdots & \ddots & \vdots \\ \Lambda_{i_2,j_1} & \cdots & \Lambda_{i_2,j_2} \end{bmatrix}.$$

However, we are interested in the image of  $D \subseteq \mathbb{F}_2^n$  by the mixing layer  $\lambda$ , so we will work with the set  $Im|_D \lambda = \{v\Lambda \mid v \in D\}$ . We can also define the image of  $D$  by only a portion  $\Lambda_{(i_1,j_1):(i_2,j_2)}$  of the mixing layer, that is  $Im|_D \Lambda_{(i_1,j_1):(i_2,j_2)} = \{v\Lambda_{(i_1,j_1):(i_2,j_2)} \mid v \in D\}$ .

The set  $Im|_D \Lambda_{(i_1,j_1):(i_2,j_2)}$  is the set obtained by projecting  $D$  on the coordinates corresponding to the boxes  $j_1, \dots, j_2$ . By type-preserving we mean to study the properties of the matrix  $\Lambda$  that makes

$$type(Im|_D \lambda) = type(D). \quad (3.10)$$

We will give a formal definition of type-preserving [\[4\]](#) matrix below;

**Definition 3.12.** A matrix  $\Lambda \in GL(\mathbb{F}_2^n)$ , which is the corresponding mixing layer  $\gamma$ , satisfying equation [\(3.10\)](#) for any  $D \subseteq \mathbb{F}_2^n$ , is called type-preserving. If  $\Lambda$  is not type-preserving, then we say that it is non-type-preserving.

We are interested  $D \subseteq \mathbb{F}_2^n$  which are subgroups of  $\mathbb{Z}_{2^n}$ , with type  $(n_w, n_r, n_b)$  satisfying the bound condition in equation [\(3.9\)](#). From now on we will consider the subsets  $D$  of  $\mathbb{F}_2^n$  that are all of this kinds. Any vector  $v \in D$  can be represented by  $v = (v_w|v_r|v_b)$  which is a concatenation of some vectors  $v_w, v_r$  and  $v_b$  whose length depend on the type  $(n_w, n_r, n_b)$  of  $D$ . In particular  $v_w \in \mathbb{F}_2^{mn_w}, v_r \in \mathbb{F}_2^{mn_r}$  and  $v_b \in \mathbb{F}_2^{mn_b}$ , from the structure of  $D$  in remark [3.1](#) have the properties below:

### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

$$\begin{cases} |\{v_w : \exists v = (v_w|v_b|v_r) \in D\}| = 1 \\ 2 \leq |\{v_r : \exists v = (v_w|v_b|v_r) \in D\}| \leq 2^{mn_r} - 1 \\ |\{v_b : \exists v = (v_w|v_b|v_r) \in D\}| = 2^{mn_b}. \end{cases} \quad (3.11)$$

Now we state and prove some lemmas. These lemmas will lead to the main result of this section, which presents the properties that the matrix  $\Lambda$  need to have to be type-preserving.

**Lemma 3.8.** *Let  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_{|D}\lambda) = (n_w, 0, \delta - n_w)$ , where  $1 \leq n_w \leq \delta - 1$ . Then*

$$\Lambda_{(n_w+1,1):(\delta,n_w)} = 0. \quad (3.12)$$

*Proof:*

We proceed by contradiction, that is, suppose that  $type(D) = (n_w, 0, \delta - n_w)$  and  $\Lambda_{(n_w+1,1):(\delta,n_w)} \neq 0$ . Now since  $v$  depends on the type of  $D$ , we consider two vectors  $v, v' \in D$  such that  $v = (v_w|v_b)$  and  $v' = (v'_w|v'_b)$  since  $n_r = 0$ . Suppose that  $v \in Ker(\Lambda_{(n_w+1,1):(\delta,n_w)})$  i.e  $v\Lambda_{(n_w+1,1):(\delta,n_w)} = 0$  and  $v' \notin Ker(\Lambda_{(n_w+1,1):(\delta,n_w)})$  then due to the properties of  $D$  in 3.11, we have that  $v_w = v'_w$ . So, if we apply  $\lambda$  on  $v$  and  $v'$  we have

$$v\Lambda = (v_w\Lambda_{(1,1):(n_w,n_w)}|0),$$

and

$$v'\Lambda = (v'_w\Lambda_{(1,1):(n_w,n_w)}|v'_b\Lambda_{(n_w+1,1):(\delta,n_w)}),$$

this means that

$$v'\Lambda = v\Lambda \oplus (0|v'_b\Lambda_{(n_w+1,1):(\delta,n_w)}),$$

here 0 denotes string of  $n_b$  zeros. Since  $v\Lambda \neq v'\Lambda$ , therefore  $\mathbf{type}(\mathbf{D}) \neq \mathbf{type}(\mathbf{Im}_{|D}\lambda)$  which is a contradiction.  $\square$

Next we check that the converse of the previous lemma holds i.e the property 3.12 implies that  $D$  and its image have the same type.

**Lemma 3.9.** *Let  $1 \leq n_w \leq \delta - 1$  and  $\Lambda_{(n_w+1,1):(\delta,n_w)} = 0$ . Then  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_{|D}\lambda) = (n_w, 0, \delta - n_w)$*

*Proof:*

We construct  $D$  with the type  $(n_w, 0, \delta - n_w)$ , then any vector  $v \in D$  can be written as  $(v_w|v_b)$  where  $v_w$  is fixed and  $v_b \in \mathbb{F}_2^{\delta - n_w}$ . Since  $\Lambda_{(n_w+1,1):(\delta,n_w)} = 0$  then for any  $v, v' \in D$ ,  $v\Lambda = (v_w\Lambda_{(1,1):(n_w,n_w)}|0)$  and  $v'\Lambda = (v_w\Lambda_{(1,1):(n_w,n_w)}|0)$  where 0 is a string of  $n_b$  zeros. We can say that the first  $mn_w$  bits of any  $v \in D$  are the same, in particular the first  $n_w$  boxes of  $Im_{|D}\lambda$  are white. Since  $\lambda$  invertible, and so  $\Lambda$  have a full rank, which is true if we assume that  $\Lambda_{(n_w+1,n_w+1):(\delta,\delta)}$  is invertible. Since  $\{v_b : \exists v = (v_w|v_b) \in D\} = \mathbb{F}_2^{\delta - n_w}$ , then  $\{v_b\Lambda_{(n_w+1,n_w+1):(\delta,\delta)} : \exists v = (v_w|v_b) \in D\} = \mathbb{F}_2^w$ , from here we can conclude that  $\mathbf{type}(\mathbf{Im}_{|D}\lambda) = (n_w, \delta - n_w, 0)$ .  $\square$

We have proved that  $\Lambda_{(n_w+1,1):(\delta,n_w)} = 0$  is a necessary and sufficient property on  $\Lambda$  to have a mixing layer which preserves the type  $(n_w, \delta - n_w, 0)$ .

The special cases, when  $type(D) = (0, \delta, 0)$  and  $type(D) = (\delta, 0, 0)$ , are trivial since all full rank matrices preserve these types.

### 3.2. TYPE-PRESERVING MATRICES

**Lemma 3.10.** *Let both  $D$  and  $Im|_D \lambda$  be of the type  $(n_w, 1, \delta - n_w - 1)$ , where  $1 \leq n_w \leq \delta - 2$ . Then  $\Lambda$  satisfies the following properties:*

- (a)  $\Lambda_{(n_w+2,1):(\delta,n_w)} = 0$ ,
- (b)  $\Lambda_{(n_w+1,1):(n_w+1,n_w)}$  is not of a full-rank matrix,
- (c)  $2 \leq |Im|_D(\Lambda_{(n_w+1,n_w+1):(\delta,n_w+1)})| < 2^{m(\delta-n_w-1)}$ ,
- (d)  $|Im|_D(\Lambda_{(n_w+1,n_w+2):(\delta,\delta)})| = 2^{m(\delta-n_w-1)}$ .

Proof:

Employing the same notation for  $v, v' \in D$ , i.e  $v = (v_w|v_b|v_r)$ . Recall that  $|\{v_w : \exists v = (v_w|v_b|v_r) \in D\}| = 1$  mean that  $v_w$  is the same for each  $v \in D$ . We prove this lemma in a sequential manner, where in each we deny each of the properties (a)-(d).

- (a) In this case we can consider  $v = (v_w|v_r|v_b)$  and  $v' = (v_w|v_r|v'_b)$  in  $D$ .  
We assume  $\Lambda_{(n_w+2,1):(\delta,n_w)} \neq 0$ , such that  $v_b \in Ker(\Lambda_{(n_w+2,1):(\delta,n_w)})$  and  $v'_b \notin Ker(\Lambda_{(n_w+2,1):(\delta,n_w)})$ , then this means that  $v\Lambda$  has different first  $n_w$  bits with respect to  $v'\Lambda$ , which contradicts the fact that  $|\{v_w : \exists v = (v_w|v_b|v_r) \in D\}| = 1$ , thus the first  $n_w$  boxes in  $Im|_D \lambda$  are in fact not white, and so violating the hypothesis of the lemma.
- (b) In this case we can consider  $v = (v_w|v_r|v_b)$  and  $v' = (v_w|v'_r|v_b)$  in  $D$ .  
We assume that  $\Lambda_{(n_w+1,1):(n_w+1,n_w)}$  is a full-rank matrix. If  $v_r \in Ker(\Lambda_{(n_w+1,1):(n_w+1,n_w)})$  and  $v'_r \notin Ker(\Lambda_{(n_w+1,1):(n_w+1,n_w)})$ , then  $v\Lambda$  and  $v'\Lambda$  have different  $m(n_w + 1)$  bits. Which means the type of  $Im|_D \lambda$  is not  $(n_w, 1, \delta - n_w - 1)$ . This is a contradiction.
- (c) Considering  $v = (v_w|v_r|v_b)$  in  $D$ .  
We assume that  $|Im|_D(\Lambda_{(n_w+1,n_w+1):(\delta,n_w+1)})| = 1$ , then if we apply  $v$  on  $\lambda$  we obtain the  $(n_w + 1)th$  term entry of the output to be a white box and not a ruled box, similarly if  $|Im|_D(\Lambda_{(n_w+1,n_w+1):(\delta,n_w+1)})| = 2^{m(\delta-n_w-1)}$  we obtain the  $(n_w + 1)th$  entry of the output to be a black box and not a ruled box. This contradicts the hypothesis of the lemma.
- (d) Considering  $v = (v_w|v_r|v_b)$  in  $D$ .  
We assume that  $|Im|_D(\Lambda_{(n_w+1,n_w+1):(\delta,n_w+1)})| < 2^{m(\delta-n_w-1)}$ , then upon applying  $v$  on  $\lambda$ , we obtain the last  $\delta - n_w - 1$  entries of the output vector to be a ruled box which contradicts the hypothesis of the lemma.

We want to see if the properties enumerated in the above lemma are not just necessary but also sufficient properties for type-preserving. □

**Lemma 3.11.** *Let  $\Lambda$  be a matrix satisfying the four properties in 3.10 then  $\lambda$  preserve the type  $(n_w, 1, \delta - n_w - 1)$ ,  $1 \leq n_w \leq \delta - 2$ .*

Proof:

If we consider  $D$  of type  $(n_w, 1, \delta - n_w - 1)$ , and its ruled box is the kernel of  $\Lambda_{(n_w+1,1):(n_w+1,n_w)}$ , then  $Im|_D \lambda$  is also of the type  $(n_w, 1, \delta - n_w - 1)$ . □

We can infer from Lemma 3.11 that the properties in Lemma 3.10 is a necessary and sufficient condition for  $\lambda$  to be type-preserving. Now we prove the special cases when  $n_w = 0$  and  $n_w = \delta - 1$ .

### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

**Lemma 3.12.** *Let  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_D \lambda) = (\delta - 1, 1, 0)$ , then  $\Lambda$  satisfies;*

- (a)  $\Lambda_{(\delta,1):(\delta,\delta-1)}$  is not a full-rank matrix,
- (b)  $\Lambda_{(\delta,\delta)} \neq 0$ .

*Conversely, if  $\Lambda$  satisfies the two properties above, then there exists  $D$  whose type is preserved by  $\Lambda$ .*

**Lemma 3.13.** *Let  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_D \lambda) = (0, 1, \delta - 1)$ , then  $\Lambda$  satisfies;*

- (c)  $2 \leq |\mathbf{Im}_D(\Lambda_{(1,1):(\delta,1)})| < 2^{m(\delta-1)}$ ,
- (d)  $|\mathbf{Im}_D(\Lambda_{(1,2):(\delta,\delta)})| = 2^{m(\delta-1)}$ .

*Conversely, if  $\Lambda$  satisfies the two properties above, then there exists  $D$  whose type is preserved by  $\Lambda$ .*

Now the consequence of these lemmas will be the next theorem which gives necessary and sufficient properties that the matrix  $\Lambda$  must possess to make it a type-preserving matrix.

**Theorem 3.1.** *The mixing layer  $\lambda$  is type-preserving with respect to the subsets of  $\mathbb{F}_2^n$  with type  $(n_w, n_r, n_b)$  satisfying equation (3.9) if and only if, there exists an integer  $n_w \in \{0, \dots, \delta\}$  for which either equation (3.13)*

$$\Lambda_{(n_w+1,1):(\delta,n_w)} = 0 \tag{3.13}$$

*or the following properties hold*

- (a)  $\Lambda_{(n_w+2,1):(\delta,n_w)} = 0$ ,
- (b)  $\Lambda_{(n_w+1,1):(n_w+1,n_w)}$  is not a full-rank matrix,
- (c)  $2 \leq |\mathbf{Im}_D(\Lambda_{(n_w+1,n_w+1):(\delta,n_w+1)})| < 2^{m(\delta-n_w-1)}$ ,
- (d)  $|\mathbf{Im}_D(\Lambda_{(n_w+1,n_w+2):(\delta,\delta)})| = 2^{m(\delta-n_w-1)}$ .

Proof:

The bounds of the  $n_w, n_b$  and  $n_r$  were given in equation (3.11), using this we have four cases to consider, namely;

1.  $\mathbf{type}(\mathbf{D}) = (0, 0, \delta)$ ,
2.  $\mathbf{type}(\mathbf{D}) = (\delta, 0, 0)$ ,
3.  $\mathbf{type}(\mathbf{D}) = (n_w, 0, \delta - n_w)$ ,
4.  $\mathbf{type}(\mathbf{D}) = (n_w, 1, \delta - n_w - 1)$ .

### 3.2. TYPE-PRESERVING MATRICES

The proof of cases 3 and 4 is the consequence of the lemmas 3.8 to lemma 3.13, cases 1 and 2 are trivial cases, since all full-rank matrices (since  $\lambda$  is an invertible linear map) preserve  $\text{type}(\mathbf{D}) = (0, 0, \delta)$  and  $\text{type}(\mathbf{D}) = (\delta, 0, 0)$ , i.e  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_{\mathbb{D}}\lambda) = (0, 0, \delta)$  and  $\mathbf{type}(\mathbf{D}) = \mathbf{type}(\mathbf{Im}_{\mathbb{D}}\lambda) = (\delta, 0, 0)$  respectively.

We will proceed to give real-life examples of ciphers that have non-type-preserving mixing layer because it does satisfy Theorem 3.1, but before proceeding to this we will give some definitions and corollaries that are consequences of Theorem 3.1.  $\square$

**Corollary 3.1.** *If  $\Lambda_{(n_w+2,1):(\delta,n_w)} \neq 0$ , for any  $n_w \in \{1, \dots, \delta - 2\}$ , then  $\Lambda$  is non-type-preserving.*

Proof:

$\Lambda_{(n_w+2,1):(\delta,n_w)}$  is a submatrix of  $\Lambda_{(n_w+1,1):(\delta,n_w)}$ . Therefore if  $\Lambda_{(n_w+2,1):(\delta,n_w)} \neq 0$  then  $\Lambda_{(n_w+1,1):(\delta,n_w)} \neq 0$  also, which violates equation (3.13) of Theorem 3.1. Thus from Theorem 3.1, if equation (3.13) and property (a) are not satisfied,  $\Lambda$  is non-type-preserving.

$\square$

**Definition 3.13.** A matrix over a finite field which has all its minors to be non-zero is called Maximum Distance Separable (MDS).

**Lemma 3.14.** *A mixing layer  $\lambda$  such that  $\Lambda$  is Maximum Distance Separable matrix is non-type-preserving.*

Proof:

Since  $\Lambda$  is MDS, then  $\Lambda_{(n_w+1,1):(\delta,n_w)} \neq 0$  for  $n_w \in \{1, \dots, \delta - 2\}$ , then from Corollary 3.1 we have that  $\Lambda$  is non-type-preserving.

#### 3.2.1. Examples of Non-type-preserving mixing layer

The aim of this subsection is to show that the mixing layer of some known block ciphers are non-type-preserving with respect to the subsets of  $\mathbb{F}_2^n$  whose type satisfies the bounds condition in equation (3.9). We considered the mixing layer of AES-like cipher[13], GOST-like cipher[3] and PRESENT cipher[7]

**Example 3.1.** *Mixing layer of an AES-like[13] cipher is the product of the ShiftRows and the MixColumns which are matrices in  $GL_{\delta}(\mathbb{F}_{2^m})$ , where  $\delta = 2^t$ , for some integer  $t$ . The ShiftRows is a circulant block matrix with entries  $\mathbb{I}_j$  for  $j \in \{1, \dots, 2^{(t/2)}\}$  where,*

$$\mathbb{I}_j = \begin{cases} 1 \in \mathbb{F}_{2^m}, & \text{position } (j, j) \\ 0 \in \mathbb{F}_{2^m}, & \text{otherwise} \end{cases}.$$

We represent the ShiftRows by its first column since it is a circulant matrix,

$$\text{ShiftRows} = \begin{bmatrix} \mathbb{I}_1 \\ \mathbb{I}_{2^{t/2}} \\ \vdots \\ \mathbb{I}_j \\ \vdots \\ \mathbb{I}_2 \end{bmatrix}. \quad (3.14)$$

### 3. PRELIMINARY RESULTS AND TYPE-PRESERVING MATRICES

The second component of the mixing layer is the *MixColumns*. It is a block matrix whose submatrices are MDS matrix in  $GL_{2^{t/2}}(\mathbb{F}_2^m)$  denoted by  $M$ ,

$$\text{MixColumns} = \begin{bmatrix} M & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & M \end{bmatrix}, \quad (3.15)$$

where  $M$  is a  $4 \times 4$  matrix in  $GL_4(\mathbb{F}_2^8)$ , and its entries are in hexadecimal

$$M = \begin{bmatrix} 2 & 1 & 1 & 3 \\ 3 & 2 & 1 & 1 \\ 1 & 3 & 2 & 1 \\ 1 & 1 & 3 & 2 \end{bmatrix}. \quad (3.16)$$

We can now define the matrix  $\Lambda_{AES}$  corresponding to the mixing layer. This is a particular case where  $\delta = 16, m = 8$

$$\Lambda_{AES} = \text{ShiftRows} \cdot \text{MixColumns} = \begin{bmatrix} M \cdot \mathbb{I}_1 & M \cdot \mathbb{I}_2 & M \cdot \mathbb{I}_3 & M \cdot \mathbb{I}_4 \\ M \cdot \mathbb{I}_4 & M \cdot \mathbb{I}_1 & M \cdot \mathbb{I}_2 & M \cdot \mathbb{I}_3 \\ M \cdot \mathbb{I}_3 & M \cdot \mathbb{I}_4 & M \cdot \mathbb{I}_1 & M \cdot \mathbb{I}_2 \\ M \cdot \mathbb{I}_2 & M \cdot \mathbb{I}_3 & M \cdot \mathbb{I}_4 & M \cdot \mathbb{I}_1 \end{bmatrix} \quad (3.17)$$

**Proposition 3.2.**  $\Lambda_{AES}$  is non-type-preserving.

Proof:

We only need show that  $\Lambda_{AES}$  is MDS, which is true since  $M$  is an MDS i.e  $\Lambda_{\delta,1} \neq 0$ . Therefore, from Lemma 3.14,  $\Lambda_{AES}$  is non-type-preserving.  $\square$

More generally, any AES-like [13] is non-type-preserving. Another example is *GPig2* [18].

**Example 3.2.** Considering any GOST-like cipher [3] we will show that this also is non-type-preserving. In the case of the GOST the needed parameters are the number of bits  $n$ , the bricks and its dimension  $\delta$  and  $m$  respectively and lastly the right rotation which is by eleven bits therefore we have  $n = 32, m = 4, \delta = 8$  and  $s = 11$ . We will represent the right rotation by 11 bits by  $\pi_{11}$  and it is given by

$$\pi_{11} = \begin{pmatrix} 1 & 2 & \cdots & 32 \\ 12 & 13 & \cdots & 11 \end{pmatrix}$$

The corresponding mixing layer is

$$\Lambda_{GOST} = \begin{bmatrix} 0 & \mathbb{1}_{21} \\ \mathbb{1}_{11} & 0 \end{bmatrix},$$

where  $0$  is an  $i \times j$  matrix and  $\mathbb{1}_i$  is a  $i \times i$  matrix, for any  $i, j \in \mathbb{N}$ .

**Proposition 3.3.** Let  $\Lambda$  be a binary circulant permutation matrix associated to the rotation of  $s$  bits. Then  $\Lambda$  is non-type-preserving if and only if  $m \leq s \leq m(\delta - 1)$ , where  $s$  is the right rotation.

### 3.2. TYPE-PRESERVING MATRICES

Proof:

We will denote the mixing layer by  $\Lambda_s$  to show the slight dependence of  $\Lambda$  on  $s$  the bits of right rotation. We

$$\Lambda_s = \begin{bmatrix} 0 & \mathbb{1}_{m\delta-s} \\ \mathbb{1}_s & 0 \end{bmatrix},$$

where  $\mathbb{1}_s$  is a  $s \times s$  identity matrix and  $\mathbb{1}_{m\delta-s}$  a  $(m\delta - s) \times (m\delta - s)$  identity matrix. We will consider for the case where  $s = m$ ,  $s > m$  and lastly when  $s$  is not in the interval  $[m, m(\delta - 1)]$ .

If  $s = m$ , then for each  $n_w \in \{1, \dots, \delta - 2\}$ , we have that  $\Lambda_{(\delta,1)} \neq 0$ , this implies that  $\Lambda_{(n_w+2,1):(\delta,n_w)} \neq 0$  therefore from the result in Corollary 3.1 we have that  $\Lambda_s$  is non-type-preserving. For  $n_w = \delta - 1$  we only need to show that  $\Lambda_s$  does not satisfy the properties in Lemma 3.13. This is true since  $\Lambda_{(1,2)} \neq 0$ . The type here is worthy of note and it is of the type  $(0, 1, \delta - 1)$ .

Now, if  $s > m$ , we used similar arguments as before i.e  $\Lambda_{(\delta,1)} \neq 0$ , for  $n_w \in \{1, \dots, \delta - 2\}$ , which means  $\Lambda_s$  is non-type-preserving from Corollary 3.1. For  $n_w = \delta - 1$ , the properties in Lemma 3.13 are not satisfied since  $\Lambda_{(n_w+1,n_w)} \neq 0$ . Thus we proved that for  $m \leq s \leq m(\delta - 1)$ ,  $\Lambda_s$  is non-type-preserving. However, for  $s$  outside the range  $[m, m(\delta - 1)]$ ,  $\Lambda_s$  is type-preserving, since if  $s = 0$  or  $s = m\delta$ ,  $\Lambda_s$  is an identity matrix that preserves the type.  $\square$

The consequence of Proposition 3.3 is the following corollary.

**Corollary 3.2.** *The mixing layer of a GOST-like cipher is non-type-preserving.*

Lastly we will consider the block cipher PRESENT[7], however with a slight modification to its mixing layer. This modification is for convenience.

**Example 3.3.** *The parameters of PRESENT are  $n = 64$ ,  $m = 4$ ,  $\delta = 16$  and  $s = 16$ . The bit permutation used is*

$$\pi_{16}(i) = \begin{cases} (16(i-1) \bmod 63) + 1 & \text{if } 1 \leq i \leq 63 \\ 64 & \text{if } i = 64. \end{cases}$$

The mixing layer associated to  $\pi_{16}(i)$  is  $\Lambda_P$  which is

$$\Lambda_{GOST} = \begin{bmatrix} 0 & \mathbb{1}_{48} \\ \mathbb{1}_{16} & 0 \end{bmatrix},$$

where  $0$  is an  $i \times j$  matrix and  $\mathbb{1}_i$  is a  $i \times i$  matrix, for any  $i, j \in \mathbb{N}$ . Now we will show that  $\Lambda_P$  is non-type-preserving.

**Lemma 3.15.** *The mixing layer of PRESENT is non-type-preserving.*

Proof:

We will prove this by employing corollary 3.1, i.e we will only show that  $(\Lambda_P)_{(n_w+2,1):(\delta,n_w)} \neq 0$  for any  $n_w \in \{1, \dots, \delta - 2\}$ , here  $\delta = 16$ . This is straightforward since the bit value 1 at positions (13, 4), (45, 12) and (61, 16) are contained in the submatrices  $(\Lambda_P)_{(3,1):(16,1)}$  and  $(\Lambda_P)_{(4,1):(16,2)}$ ,  $(\Lambda_P)_{(n_w+2,1):(16,n_w)}$  for any  $n_w \in \{3, \dots, 10\}$  and  $(\Lambda_P)_{(n_w+2,1):(16,n_w)}$  for any  $n_w \in \{11, \dots, 14\}$  respectively. This in turn means that  $(\Lambda_P)_{(n_w+2,1):(\delta,n_w)} \neq 0$  for any  $n_w \in \{1, \dots, 14\}$ .  $\square$



## 4. Applications

Block ciphers are designed according to either the Feistel Network (FN) or the Substitution Permutation Network (SPN), in this chapter we will discuss chiefly the effect of a non-type-preserving mixing layer on the imprimitivity of the group generated by the round functions of a block cipher. We will proceed by considering a generalized block cipher that is designed according to SPN and then FN. We considered SPN with addition modulo  $2^n$  key mixing which we will denote henceforth as SPNmod and as for FN we considered GOST-like cipher.

### 4.1. PRIMITIVITY OF AN SPNMOD CIPHER

The goal of this section is to prove the primitivity of a generalized SPNmod cipher under the assumption that the mixing layer is non-type-preserving and the S-Boxes are invertible. We define the bricks of the message from the plaintext space below;

$$V = V_1 \times V_2 \times \cdots \times V_\delta, \quad (4.1)$$

where the dimension of each brick is  $m$ , i.e  $\dim(V_i) = m, \forall i \in \{1, \dots, \delta\}$

**Theorem 4.1.** *Let  $\mathcal{C}$  be an SPNmod cipher acting on the plaintext space  $V$ , in which the round function is of the form defined in equation (2.2),*

$$\varepsilon_{k_h} = \gamma \lambda \sigma_{k_h},$$

for the each round key  $k \in \mathcal{K}$ , the key space, where  $\gamma, \lambda, \sigma_{k_h}$  still have the same meaning, but for convenience we state again their meaning;

- $\gamma : V \rightarrow V$  is a non-linear permutation, called parallel S boxes which acts in parallel on each bricks  $V_j$ , i.e

$$(x_1, x_2, \dots, x_n) \gamma = ((x_1, \dots, x_m) \gamma_1, \dots, (x_1, \dots, x_m) \gamma_\delta),$$

where  $\gamma_i \in \text{Sym}(V_i)$  and  $0 \gamma_i \neq 0$ .

- $\lambda \in \text{Sym}(V)$  is a linear map called mixing layer and it is non-type-preserving.
- $\sigma_k : V \rightarrow V$  is the  $\boxplus$  – translation of  $V$  by  $k$ , i.e  $\forall v \in V \ v \sigma_k = v \boxplus k$ .

Then  $\Gamma_\infty = \Gamma_\infty(\mathcal{C})$  is primitive.

Proof:

We start this prove by giving the definition of a primitive group. The group  $\Gamma_\infty = \langle \mathcal{T}, \gamma \lambda \rangle$  see equation (3.7), is primitive if;

- its action on  $V$  is transitive
- There's no block system.

## 4.2. PRIMITIVITY OF GOST-LIKE CIPHERS

Lemma 3.4 proves that  $\Gamma_\infty = \langle \mathcal{T}, \gamma\lambda \rangle$  is transitive on  $V$ , for convenience we denote  $\gamma\lambda$  by  $\kappa$ . Thus we need only to show that there is no block system. In order to prove this we employ Lemma 3.6, i.e we need to show that there is no non-trivial subgroup  $D$  that satisfies Lemma 3.6, which means

$$D\kappa = v \boxplus D$$

only if  $D$  is a trivial proper subgroup of  $(V, \boxplus)$ . Let  $0$  be the string of  $n$  zeros, then  $0 \in D$  and  $v \boxplus 0 = v$  this means that

$$v = 0\kappa.$$

So we need only to prove that for each non-trivial proper subgroup  $D$  of  $V$  we have  $D\kappa \neq 0\kappa \boxplus D$ . The S-Boxes are parallel and invertible, thus they map any set with a type to another set with the same type, this means that  $D\gamma$  and  $D$  has the same type. Also  $v \boxplus D$  has the same type as  $D$  from Lemma 3.7. This means that  $v \boxplus D$  and  $D\gamma$  have the same type. But  $\lambda$  is a non-type-preserving mixing layer therefore  $D\gamma\lambda$  doesn't have the same type as  $D\gamma$ , which implies that

$$D\kappa \neq v \boxplus D,$$

for every non-trivial proper subgroup  $D$  of  $V$ . This means that  $\Gamma_\infty$  is primitive.  $\square$

**Remark 4.1.** *The cipher Gpig2[18] has all the properties of the Theorem 4.1 and it is SPNmod, so the group generated by its round functions is primitive.*

## 4.2. PRIMITIVITY OF GOST-LIKE CIPHERS

GOST-like ciphers are designed according to Feistel Networks, thus the message is partitioned into 2 halves, using this information we represent the plaintext space by Cartesian product of these halves i.e  $V = V^1 \times V^2$ , where  $V^i$ ,  $i = 1, 2$ , is  $\mathbb{F}_2^n$ . The mixing layer is any non-type-preserving matrix instead of a rotation. We then prove that under the assumptions given below, the group generated by this round functions is primitive.

We consider some assumptions on a GOST-like cipher.

- The plaintext space  $V = V^1 \times V^2$ , and  $V^i$ ,  $i = 1, 2$  are further splitted into bricks, i.e

$$V^i = V_1^i \times V_2^i \times \dots \times V_\delta^i,$$

where  $m > 1$  is the dimension of each brick and  $\delta > 1$  is the number of the bricks.

- The parallel S-Box which is a non-linear invertible map  $\gamma \in \text{Sym}(V^i)$  that acts in parallel on each brick  $V_j^i$  and  $0\gamma_j = 0$ , where  $\gamma_j \in \text{Sym}(V_j^i)$ .
- The mixing layer  $\lambda \in \text{Sym}(V^i)$  that is non-type preserving.
- Key mixing  $\sigma : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ , which is a  $\boxplus$ -translation.
- $\kappa = \gamma\lambda \in \text{Sym}(V^i)$ .

Moreover, we consider the key mixing  $\sigma$  using the subkeys  $(k_1, k_2) \in \mathcal{K} \times \mathcal{K}$  on  $(x_1, x_2) \in V^1 \times V^2$ , i.e

$$(x_1, x_2)\sigma_{(k_1, k_2)} = (x_1 \boxplus k_1, x_2 \boxplus k_2)$$

To introduce the Feistel Network (FN) structure, we employ a formal matrix  $\mathcal{P}$  of dimension  $2n \times 2n$ .

$$\mathcal{P} = \begin{bmatrix} 0 & 1 \\ 1 & \kappa \end{bmatrix},$$

where each of the entries in  $\mathcal{P}$  are  $n \times n$  matrix. For any  $(x_1, x_2) \in V^1 \times V^2$ , we have the right action of  $\mathcal{P}$  on  $(x_1, x_2)$

$$\begin{aligned} (x_1, x_2)\mathcal{P} &= (x_1, x_2) \begin{bmatrix} 0 & 1 \\ 1 & \kappa \end{bmatrix} \\ &= (x_2, x_1 \oplus x_2\kappa). \end{aligned} \tag{4.2}$$

We can now define the group generated by the round function of any GOST-like cipher

$$\Gamma_\infty = \langle \sigma_k \mathcal{P} \sigma_h : k, h \in \mathcal{H} = \mathcal{K} \times \mathcal{K} \rangle$$

**Theorem 4.2.** *Let  $\mathcal{C}$  be a generalized GOST-like cipher as defined above. If the parallel S-Box  $\gamma$  is in  $\text{Sym}(V^i)$ , then  $\Gamma_\infty(\mathcal{C})$  is primitive.*

Proof:

Lemma 4.7 in Section 4 of [3] proves this using Goursat's Lemma[15], the case when  $D\kappa = 0\kappa \boxplus D$ . We then consider this case  $D\kappa = 0\kappa \boxplus D$  with  $D$  a non-trivial proper subgroup of  $\mathbb{Z}_{2^n}$ , in the same way we approached it in the proof of Theorem 4.1, and we apply Theorem 3.1, then we conclude that  $\Gamma_\infty(\mathcal{C})$  is primitive.

## 5. Conclusion

This thesis focused on imprimitivity attacks [paterson1999imprimitive] on block ciphers, since block cipher are known for their strength in resisting other known attacks namely, linear, differential and algebraic cryptanalysis attacks. Our main result is on the mixing layer, we considered the binary matrix associated with the mixing layer. On this binary matrix we presented the conditions necessary and sufficient for it to be type-preserving and we said that if this type-preserving property is lacking then the binary matrix is called is said to be non-type-preserving.

We then showed in theorems 4.1 and 4.2 that if the binary matrix corresponding to the mixing layer is non-type-preserving then the resistance to imprimitivity attacks is guaranteed. It is worthy of mention that the block cipher we considered is the one whose key mixing is by addition modulo  $2^n$ , where  $n$  is the dimension of the plaintext.

A possible future research direction will be to consider a GOST-like cipher with a key-schedule, a non-type-preserving mixing layer and a parallel S-Box. This will lead to a more detail analysis of the security of the cipher which will include the study of statistical attack on this type of GOST-like cipher. This should give a better apprehension of the inner workings of the cipher which uses key mixing function of addition mod  $2^n$ .

# Bibliography

- [1] A brief history of cryptography august 14,. <https://www.redhat.com/en/blog/brief-history-cryptography#:~:text=The%20firs%20known%20evidence%20of,place%20of%20more%20ordinary%20ones>.
- [2] Hoda A Alkhzaimi. Cryptanalysis of selected block ciphers. 2016.
- [3] Riccardo Aragona, Andrea Caranti, and Massimiliano Sala. The group generated by the round functions of a gost-like cipher. *Annali di Matematica Pura ed Applicata (1923-)*, 196(1):1–17, 2017.
- [4] Riccardo Aragona and Alessio Meneghetti. Type-preserving matrices and security of block ciphers. *arXiv preprint arXiv:1803.00965*, 2018.
- [5] Gregory Bard. *Algebraic cryptanalysis*. Springer Science & Business Media, 2009.
- [6] Eli Biham and Adi Shamir. *Differential cryptanalysis of the data encryption standard*. Springer Science & Business Media, 2012.
- [7] Andrey Bogdanov, Lars R Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte VIKKELSOE. Present: An ultra-lightweight block cipher. pages 450–466, 2007.
- [8] Mohammad Ubaidullah Bokhari, Shadab Alam, and Faheem Syeed Masoodi. Cryptanalysis techniques for stream cipher: a survey. *International Journal of Computer Applications*, 60(9), 2012.
- [9] Marco Calderini and Massimiliano Sala. Elementary abelian regular subgroups as hidden sums for cryptographic trapdoors. *arXiv preprint arXiv:1702.00581*, 2017.
- [10] Peter J Cameron et al. *Permutation groups*. Number 45. Cambridge University Press, 1999.
- [11] C Charnes, L O’Connor, J Pieprzyk, R Safavi-Naini, and Y Zheng. Soviet encryption algorithm. 1998.
- [12] Keith Conrad. Transitive group actions. *University of Connecticut, Department of Mathematics*, 2009.
- [13] Joan Daemen and Vincent Rijmen. The rijndael block cipher: Aes proposal. pages 343–348, 1999.
- [14] Vasily Dolmatov. Gost 28147-89: Encryption, decryption, and message authentication code (mac) algorithms. Technical report, 2010.
- [15] Edouard Goursat. Sur les substitutions orthogonales et les divisions régulières de l’espace. In *Annales scientifiques de l’École Normale Supérieure*, volume 6, pages 9–102, 1889.
- [16] Howard M Heys. A tutorial on linear and differential cryptanalysis. *Cryptologia*, 26(3):189–221, 2002.

## BIBLIOGRAPHY

- [17] Burton S Kaliski, Ronald L Rivest, and Alan T Sherman. Is the data encryption standard a group?(results of cycling experiments on des). *Journal of Cryptology*, 1(1):3–36, 1988.
- [18] Debdeep Mukhopadhyay and Dipanwita RoyChowdhury. Key mixing in block ciphers through addition modulo  $2^n$ . *Cryptology EPrint Archive*, 2005.
- [19] Kenneth G Paterson. Imprimitve permutation groups and trapdoors in iterated block ciphers. In *International Workshop on Fast Software Encryption*, pages 201–214. Springer, 1999.
- [20] Claude E Shannon. Communication theory of secrecy systems. *The Bell system technical journal*, 28(4):656–715, 1949.
- [21] Nigel Paul Smart et al. *Cryptography: an introduction*, volume 3. McGraw-Hill New York, 2003.
- [22] Arthur Sorkin. Lucifer, a cryptographic algorithm. *Cryptologia*, 8(1):22–42, 1984.