CAPITAL UNIVERSITY OF SCIENCE AND
TECHNOLOGY, ISLAMABAD

A MATLAB Tool for the Analysis of Cryptographic Properties of S-boxes

by

Fatima Ishfaq

A thesis submitted in partial fulfillment for the
degree of Master of Philosophy

in the

Faculty of Computing
Department of Mathematics

2018

To my father for his prayers and love, for giving me the determination to overcome many trying moments to pursue my dreams.

CAPITAL UNIVERSITY OF SCIENCE & TECHNOLOGY
ISLAMABAD

CERTIFICATE OF APPROVAL

A MATLAB Tool for the Analysis of Cryptographic Properties of S-boxes

by

Fatima Ishfaq
MMT153016

THESIS EXAMINING COMMITTEE

S. No.	Examiner	Name	Organization
(a)	External Examiner	Dr. Umar Hayat	QAU, Islamabad
(a)	Internal Examiner	Dr. Dure Shehwar Sagheer	CUST, Islamabad
(b)	Supervisor	Dr. Rashid Ali	CUST, Islamabad

Dr. Rashid Ali
Thesis Supervisor
September, 2018

Dr. Muhammad Sagheer
Head
Dept. of Mathematics
September, 2018

Dr. Muhammad Abdul Qadir
Dean
Faculty of Computing
September, 2018

Author's Declaration

I, **Fatima Ishfaq** hereby state that my M.Phil thesis titled "**A MATLAB Tool for the Analysis of Cryptographic Properties of S-boxes**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad.

At any time if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my M.Phil Degree.

(Fatima Ishfaq)

Registration No: MMT-153016

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "*A MATLAB Tool for the Analysis of Cryptographic Properties of S-boxes*" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been dully acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of M.Phil Degree, the University reserves the right to withdraw/revoke my M.Phil degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Fatima Ishfaq)

Registration No: MMT-153016

Acknowledgements

All praise be to **Almighty ALLAH** who has been bestowing me with His great bounties and enabled me to complete my dissertation.

I would like to thank my affectionate teachers, Dr. Muhammad Sagheer, Dr. Abdul Rehman Kashif, Dr. Shafqat Hussain, Dr. M. Afzal and Dr. Rashid Ali for their excellent teaching and support during these years. I would like to express my special gratitude to my supervisor **Dr. Rashid Ali** for his patience with me and guidance. A more supportive and considerate supervisor I could not have asked for. He was a big support and motivation in the difficult times as he encouraged and helped a lot during research and writing of thesis. I feel really blessed and proud to be his student.

I truly appreciate my mother for her love and continuous support, for without her, I would not have finished my degree. I am grateful to my late father for all the prayers, who always supported me in achieving my targets. I would like to thank my sister for the motivation and my kids who are always been a sweet delight of life and helped me to be a better human.

I have also had the good fortune to study with some wonderful people: Saba Majeed, Urwa Aftab, Afsheen Nazar, Maria Qamar and Sumaira Bibi who helped and encouraged me throughout my studies. Their friendships are what I will miss the most and hope to keep forever. Especially, I would like to acknowledge Ms. Saba Majeed and appreciate her friendship and contribution. My friend Zareen, your support made all the difference and I can't thank you enough for driving me towards my goal always.

Finally, I am obliged to all the people who prayed for me, shared their knowledge during my degree program and supported me.

"Once dust, you are now spirit. Once ignorant, now wise. He who has led you so far, will guide you further."

Rumi

Abstract

Substitution boxes (S-boxes) are one amongst the crucial parts within a block cipher and play an important role in their security for being the only non-linear part of the system. Despite the fact that some old algorithms and S-boxes are still in use these days as they possess theoretically known security parameters, there is always a demand for a tool which can investigate the strength of an S-box and its corresponding boolean functions cryptographic properties. A well built tool for the analysis is also necessary when we want to create new S-boxes. Although in some cases, the common properties of S-boxes are known because of their generating algorithms but still we believe that an extensive investigation is required for all cryptographic properties. In this thesis, we present a tool for the evaluation of some key properties of Boolean functions and S-boxes suitable for cryptography. The performance indexes of S-boxes are outlined and surveyed. The corresponding numerical formulas for calculating them are discussed. Then, a code is established in MATLAB software which can not only compute the required parameters for an S-box but also for the boolean functions with large computational space $GF(2^n)$ for $2 \leq n \leq 20$. This tool provides an in depth approach of calculation process for the user so we believe that this tool is very useful to support the researchers for the analysis and designing of S-boxes to check the security of associated symmetric encryption scheme.

Contents

List of Figures

List of Tables

Abbreviations

DES	Data Encryption Standard
AES	Advanced Encryption Standard
SPN	Substitution Permutation Network
SAC	Strict Avalanche Criterion
WHT	Walsh Hadamard Transform
SET	S-box Evaluation Tool
GF	Galois Field
TT	Truth Table
ANF	Algebraic Normal Form
BIC	Bit Independence Criterion
DD	Dynamic Distance

Symbols

$\mathbb{G}$	Group
$\mathbb{Z}$	Set of integers
$\mathbb{R}$	Set of real numbers
$\mathbb{Q}$	Rational numbers
$\mathbb{C}$	Complex numbers
$\mathbb{V}$	Vector space
$\mathbb{G}^*$	Multiplicative Group
$p,\ q$	Prime number
$\mathbb{F}_p$,$\mathbb{Z}_p$	Finite field of order prime p
$\mathbb{F}_{p^k}$	Finite field extension

Chapter 1

INTRODUCTION

Cryptography is the science of special techniques of secret writing which changes original message into a coded message with an unreadable format for its transmission over the public networks in the presence of adversaries.

For this purpose, we need a system or procedure for converting data or messages into secret codes. Such systems are known as **Cryptosystems**. These systems are considered usually as mathematical techniques with specific computer programs; however, they also take into account the management of human behavior e.g choosing passwords which are hard-to-guess, shut down the systems which are unused, and of course not discussing sensitive procedures with outsiders. Thus a good cryptosystems should meet Confidentiality, Integrity, Non Repudiation and Authentication criteria.

A typical cryptosystem has five major components which are Planitext, Ciphertext, Encryption Algorithm, Decryption Algorithm and Key.

Now if we consider the design criteria of a cryptosystem, the cryptography is further divided into following two main categories:

1. **Symmetric key cryptosystem**
2. **Asymmetric key cryptosystem**

Symmetric key cryptosystem in which sender and receiver share a common secret key for both data encryption and decryption. It is also known as the secret key cryptosystem. Examples are (DES) Data Encryption Standard [16] and (AES) Advanced Encryption Standard [42]. We can consider merits of symmetric key cryptosystem as simple to use, easy to implement and fast speed while demerits are key management and security issues.

For enhancing cryptosystem's security, in 1976 White Field Diffie and Martin Helman [42] proposed the idea for encrytion and decryption of data known as Asymmetric key cryptosystem or public key cryptosystem. This cryptosystem uses two keys, the key used for data encryption is different from the key used for data decryption. Thus the encryption key is shared publicly, known as Public Key and the decryption key is kept secret by the owner, known as Private Key. Examples are RSA cryptosystem [43], El.Gamal cryptosystem [18], Elliptic curve cryptosystem [7].

1.1 Substitution Boxes in Cryptography

A symmetric key cryptosystem is further categorized as either by stream cipher or block cipher. A block cipher will convert a whole block of plaintext into a block of ciphertext using the secret key at a time whereas stream cipher encrypts one bit or byte of data at a time. Thus a block cipher has two basic parameters, Block size and Key size.

The block ciphers are designed on the basis of Shannon's theory of confusion and diffusion which is also implemented in Substitution-Permutation networks(SPN) [28]. Such networks are basically consist of a number of mathematical operations which are linked together. It takes as input a block of palintext, a key and apply many rounds of **Substitution-Box** (S-box) or **Permutation-Box** (P-Box) to get desired cipher text. For decryption process the inverse S-box or P-box is used in reverse order with the same key. The examples of SPN are the Data

Encryption Standard (DES) [16] and Advanced Encryption Standard (AES) [42] cryptosystems.

S-boxes are basically vectorial boolean functions expressed as look-up tables. An S-box takes in a small block of bits and substitute them by another block of bits. This substitution should be one-to-one to make decryption effective. Generally, S-box takes m input bits and convert them into n output bits. So an $(m \times n)$ S-box can be considered as a look-up table with 2^m words of n bits each. The length of the output can be the same as the length of the input as in AES or it can be different as in DES. An S-box should be designed in such a way that each output bit will depend on every input bit for making cryptosystem strong.

1.2 Why We Study S-boxes?

The only nonlinear part of a SPN as a cryptosystem is the S-box because S-boxes are composed of highly nonlinear Boolean functions. Without them, adversaries would compromise the system with ease.

Why should we allocate time to study S-boxes while we can use them directly?

Actually, there are three main reasons for studying the S-box design.

1. **Critical to Block Ciphers**

   If you are not studying S-box design criteria then you are bound to adopt an important part of block ciphers just like a black box with no real understanding of what is their design and how it is affecting the whole system.

2. **Designing New Ciphers**

   For designing a new cipher, S-box design is the most significant area because it is the only nonlinear part of the system. So basically a cipher strength depends on this part. As with advancement of cryptography, hackers are

also developing new methods of attacks, so S-box design should be secured in advance to guarantee cipher security.

3. **Need of Developing Private S-boxes**

   Interest and awareness in this topic was increased especially when back-doors are used by the adversaries to generate keys for certain ciphers such as AES [42], therefore, every organization and especially governments want to have a secure system only applicable to their organization with an extra security layer which is possible only if they design their individual S-boxes for their specific system.

1.2.1 Classification of S-boxes

There are three sub classifications of S-boxes.

1. **Straight S-box**

   A straight S-box takes input and gives output of the same size. The well known AES [42] is an example of such S-box. This is the easiest and most common form of S-box design.

2. **Compressed S-box**

   An S-box which takes in more input bits but puts out less bits. DES is a good example of this type of S-box in which each block takes in 6 bits and outputs 4 bits block.

3. **Expanded S-box**

   This S-box takes in less input bits and puts out more bits. One can construct such S-boxes by duplicating some of the output or input bits.

1.3 Construction of S-boxes

There are many defined methods for making good S-boxes. Some examples are twofish [45], DES [16], AES [42], and GOST [13] etc.

Researchers and Cryptographers proposed many approaches and methods for the construction of a strong S-box. The security arguments of symmetric encryption algorithms are basically depending on the properties of S-boxes and so they are really crucial in cryptography.

In this scenario a main question arises that, can we suggest some S-boxes are better than others?. Obviously answer is **Yes**, so the main focus was to investigate those measures which would differentiate between **bad** and **good** substitutions, and for those techniques which would construct **good** substitutions. Cryptanalysis attacks depend on the weakness of cryptosystem so new attacks produce a demand for the new security parameters.

1.3.1 History of Design Criteria of S-boxes

We consider briefly some key points in the evolution of design criteria of good S-boxes.

1. **Feistel, H.** [19] introduced the Avalanche effect. If we had a box with 128 input and output bits, any analyst than need to deal with 2^{128} or more than 10^{38} possible digit blocks for frequency analysis of system. As the input moved through successive layers, the pattern of 1's generated was improved and resulted in an uncertain avalanche. In the end the final output had, on average, half 0's and half 1's, so all inputs were potentially involved in all outputs.

2. **Kam, J. and Davida** [26] gave the idea of Completeness. They were interested in specific structure called Substitution-Permutation (S-P) cipher. "**Completeness** of a cryptographic transformation hold if each cipher text bit must depend on all of the output bits".

3. **Gordon and Retkincount** [21] presented the randomly chosen S-boxes which had linear relationships.

4. **Webster and Tavares** [54] worked again on the concept of avalanche and completeness of S-boxes and gave us Strict Avalanche Criterion (SAC).
   "**Avalanche effect** was defined as an average of one half of the output bits should change whenever a single input bit was changed in system".
   Strict Avalanche criterion was defined as "each output bit should change with a probability of one half whenever a single input bit is complemented".

5. **Pieprzyk and Finkelstein** [39] debated first time on the non-linearity of randomly chosen S-boxes.

6. **Forre** [20] connected Walsh Spectrum with SAC for making the calculations easier.

7. **Meier and Staffelbach** [32] formulated the concept of **Perfect Non-linearity** and then narrated this as an important factor of diffusion process which creates SAC.

8. **Pieprzyk and Finkelstein** [39] worked on the the design and construction of non-linear permutations which generate S-boxes.

9. **Lloyd** [41] investigated the relationship between some important criteria of S-boxes such as SAC, Balance, and Correlation Immunity.

10. **Adams** [1] suggested the use of bent function in S-boxes to get high non-linearity and other criteria.

11. **Daemen et.al** [15] presented **The correlation matrix of a boolean mapping** which was supposed to be the best natural demonstration for the right understanding and explanation of linear cryptanalysis technique and procedure.

12. **Youssef and Tavares** [57] not only gave the expected probability of selecting an affine S-box out of a wide range of S-boxes but also argued about the leakage of those particular facts and figures which contributed when we chose functions randomly.

13. **Zhang and Zheng** [46] studied SAC and Propagation criterion. They merged these concepts and introduced a **Global Avalanche Criterion(GAC)**.

1.4 Desirable Properties for Good S-boxes

The desirable properties of an S-box are its design simplicity, fast encryption and decryption speed and resistance against known cryptanalysis attacks. The criteria of a good S-box will encounter most of the standards set by the National Institute of Standards and Technology. Some important are

1. S-box is Balanced.
2. S-box has high Non-Linearity.
3. All linear combinations of S-box columns are bent.
4. All entries in the XOR table are 0 or 2.
5. S-box satisfies Bit Independence Criteria.
6. S-box satisfies Strict Avalanche Criteria.
7. S-box has Correlation Immunity.
8. The set of weights of rows has a binomial distribution with mean $n/2$. [17]
9. Each column has hamming weight 2^{n-1}.

Nevertheless, it is impossible to achieve all criteria to their best in a single S-box. Their disagreeing nature limits us to compromise some of the criteria. For example, correlation immunity conflicts with high non-linearity and maximum non-linearity also conflicts with balance. It will depend on the applied problem that which criteria we want to achieve and on which we can negotiate.

1.5 Software Tools For S-box Analysis

For studying the properties of S-box, some tools are available. A brief description of such tools is given below:

1. **Boolfun Package in *R***

   R is the free open source Mathematical software used for computing statistics. It works on various Windows, UNIX and Mac OS platforms, while the standard version of ***R*** does not support the evaluation of Boolean functions but it is possible to load a package named as **Boolfun** [29] which provides functionality related to the cryptographic analysis of Boolean functions.

2. **SageMath**

   SageMath library [49] is the free open source Mathematics tool which contains a module called Boolean functions and an S-box. With this tool we can check the algebraic properties and calculate different cryptographic properties related to linear approximation matrix and difference distribution table for S-boxes and Boolean functions.

3. **VBF**

   VBF stands for Vector Boolean Function Library. Alverez-Cubero and Zuffiria [3] presented this tool for the analysis of vector boolean functions that are used to evaluate the cryptographic properties of S-boxes.

4. **SET**

   Stjepan Picek [37] and team presented this tool for the evaluation of cryptographic properties of Boolean function and S-boxes. SET stands for S-box Evaluation Tool. It is a free open source Mathematics tool which is simple and easy to use. It works in VS(visual studio).

1.6 Thesis Objective

As S-box plays a significant role in SPN cryptography so it is imperative to check different properties of an S-box which are used to determine the strength of system. An S-box is constructed strongly to gain protection against linear and differential crypt-analysis attacks. Non-linearity, Low number of fixed points and opposite fixed points, high algebraic degree, SAC and BIC are considered as important properties of a good S-box.

Different tools and techniques are available to check the desired properties of S-boxes but all other software are either not easily available or the user is unable to investigate the calculation of required parameters for every boolean function also. The available tools do not provide an in depth approach for the evaluation of non-linearity of an S-box. Hence user can not understand and check that how by changing the input entries of an S-box, alteration of results at different steps will transpire through boolean functions. For example, if we change one input entry in some considered S-box, by this tool we can check the change in dynamic distance at all entries of the S-box.

S-box Analysis MATLAB Tool (SAMT) is established in MATLAB language which provides a great space for efficient calculation and relaxed programming with many built in commands for user ease.

SAMT is developed for the analysis of cryptographic properties of boolean functions and the S-box providing a large computational space in $GF(2^n)$ for $2 \leq n \leq 20$. The properties which can be evaluated are:

1. Fixed points of S-box
2. Opposite Fixed points of S-box
3. Bijective
4. Hamming weight of all boolean functions
5. Balanced

6. Non-linearity of all boolean functions
7. Average non-linearity of all boolean functions
8. Almost Bent non-linearity value
9. Perfect non-linearity value of boolean functions
10. Non-linearity of S-box
11. Maximum non-linearity value for given Galois field
12. Differential Branch Number
13. Dynamic Distance Table
14. Dynamic Distances of Boolean functions
15. Desired Avalanche value for boolean function
16. Avalanche value of Boolean functions
17. Boolean functions results which satisfy Avalanche Effect
18. Avalanche Effect Percentage of S-box
19. Dependence Matrix of S-box
20. Desired SAC of a boolean function
21. SAC values of boolean functions
22. Autocorrelation function of all boolean functions
23. Sum of Squares Indicator of boolean functions
24. Absolute Indicator of boolean functions
25. BIC non-linearity criterion of S-box

Some parameters are already defined for user consideration and guidance in SAMT, e.g the desired bent value for boolean functions, maximum non-linearity value of an S-box in given Galois field and desired Avalanche value for a boolean function.

All properties calculated by this tool are labeled properly for user understanding. User can check the calculated values of every important property of boolean functions also which are basically constructing the S-box.

User can request for this MATLAB tool file on following emails.

rashid.ali@cust.edu.pk
fatimaishfaq88@gmail.com

The rest of the thesis is systematized as follows

- **Chapter 1** provides necessary information on the S-boxes importance in cryptography, their structure, construction, design criteria and softwares available for their properties analysis.
- **Chapter 2** We explore more in the context of Cryptography, introduce the Galois field, boolean functions, their general properties and how they are responsible for the construction of strong S-boxes. Different cryptographic properties were also explained according to the general design criteria of S-boxes. We then reviewed the scheme for the construction of S-box of AES with primitive polynomial $x^8 + x^4 + x^3 + x + 1$ because we considered this S-box as a basic example for the designed tool verification.
- **Chapter 3** We presented the tool in MATLAB which was basically designed for the calculation of different desired properties of an S-box according to linear and differential cryptanalysis such as non-linearity, fixed points, SAC and BIC etc. AES S-box and some other S-boxes were checked for their properties by this tool and the results were presented.

  Finally we suggest some future work that could be done to improve this tool for evaluation of more properties of S-boxes.

Chapter 2

S-boxes and their Cryptographic Properties

In this chapter we introduce and explain the basic definitions from group theory that will be used in the understanding of S-Boxes such as Galois fields and Boolean functions. Then we include some important crytographic properties of S-boxes which are considered inevitable for their strength analysis.

2.1 Mathematical Background

First some basic concepts of group theory are presented to understand the reason behind the construction and performance of the S-boxes.

2.1.1 Galois Field

A field consists of a finite number of elements is called Galois field or finite field with order as a prime or a power of prime. For a given prime number q, the set of integers $\{0, 1, 2, 3, ...q-1\}$ forms a field of order q so is called prime field $GF(q)$ and also the field of residue classes modulo q. For two elements $a, b \in GF(q)$, $a = b$ in $GF(q)$ means the same as $a \equiv b \bmod q$.

An extension of a prime field is called an Extended finite field or Galois field $GF(q^n)$ with n a positive integer number [34].

From the cryptographic point of view, we mostly focused in the cases:

- $GF(q), \quad n = 1$
- $GF(q^n), \quad q = 2$

Definition 2.1.1 (Polynomial over $GF(q)$)
An expression which consists of variables and coefficients that satisfy certain operations is called a **Polynomial**. All the elements of a finite field can be written in the form of polynomials of degree less than n, with coefficients from $GF(q)$ as

$$a_1 + a_2x + a_3x^2 + ... + a_nx^{n-1}$$

or by vector

$$[a_1 \quad a_2 \quad a_3 \quad ..... \quad a_n]$$

where a_i are the coefficients from $GF(q)$, x is its variable also known as indeterminate. The degree of polynomial is the highest power of x with largest value of n such that $a_n \neq 0$.

Definition 2.1.2 (Irreducible Polynomial)
A polynomial $m(x)$ with integer coefficients is said to be irreducible if it is an irreducible element of the polynomial ring, means it cannot be factorized as a product of two polynomials of lower degree. Otherwise it is called reducible polynomial [25].

Example 2.1.3 The polynomials $x^2 + 1$, $x^2 + x$ are reducible polynomials over $GF(2)$ and $x^2 + x + 1$, $x^3 + x + 1$ are irreducible polynomials over $GF(2)$ [27]. There are 30 irreducible polynomials of degree 8 with coefficient in $GF(2)$.

Irreducible polynomials are important as polynomial multiplication in $GF(q^n)$ is performed over modulo an irreducible polynomial.

Definition 2.1.4 (Primitive Polynomial)
An irreducible polynomial of degree n over $GF(q)$ is said to be a primitive polynomial that divides any $a(x) = x^m + 1$ where $m = q^n - 1$, but not divides any such $a(x)$ with smaller m [4].

There exist n different roots of a primitive polynomial with degree n in $GF(q^n)$ where the order of all roots is $q^n - 1$, therefore, if r is such a root, then $r^{q^n-1} = 1$.

Example 2.1.5 The polynomial $m(x) = x^3 + x + 1$ is a primitive polynomial of degree 3 in $GF(2^3)$. If there exists a smallest positive integer $t = 7$ such that $m(x) = x^3 + x + 1$ divides $x^t - 1 = x^7 + 1$ as

$$x^7 + 1 = (x^3 + x + 1)(x^4 + x^2 + x + 1)$$

So if r is the root of $x^3 + x + 1$, then $r^7 = 1$. The powers of r in $GF(2^3)$ are given in the table below:

Decimal	Roots	polynomials
0	r^0	1
1	r^1	r
2	r^2	r^2
3	r^3	$r+1$
4	r^4	r^2+r
5	r^5	$r+1+r^2$
6	r^6	r^2+1
7	r^7	1

TABLE 2.1: Roots of primitive polynomial in $GF(2^3)$

Thus different powers of r can represent all the elements of $GF(2^3)$.

2.1.1.1 Representation of Galois field elements

The elements of $GF(q^n)$ can be represented by polynomials of degree less than n with coefficients from $GF(q)$.

Example 2.1.6 Finite field $GF(2^8)$ consists of 256 elements and is used in advanced encryption standard (AES) which was created by using a fixed irreducible polynomial $m(u) = u^8 + u^4 + u^3 + u + 1$.

Thus each element of $GF(2^8)$ is a polynomial of defree less than 8. The multiplication of polynomials is reduced modulo $m(u)$.

Equivalently these elements can be represented by 8-bit binary number or 2-digit hexadecimal numbers or a positive integer from 0 to 255 inclusive.

For the finite field $GF(2^8)$, both the polynomial and binary representations are given below:

Decimal	Polynomial	Binary	Hexadecimal
0	0	00000000	00
1	1	00000001	01
2	u	00000010	02
3	$u+1$	00000011	03
4	u^2	00000100	04
5	u^2+1	00000101	05
6	u^2+u	00000110	06
7	u^2+u+1	00000111	07
8	u^3	00001000	08
9	u^3+1	00001001	09
10	u^3+u	00001010	$0A$
.	.	.	.
.	.	.	.
.	.	.	.
255	$u^7+u^6+u^5+u^4+u^3+u^2+u+1$	11111111	FF

TABLE 2.2: Elements of Finite Field $GF(2^8)$ with irreducible polynomial $m(u) = u^8+u^4+u^3+u+1$

2.1.2 Boolean Functions

A function $f : GF(2^n) \to GF(2)$ is called a Boolean function if it has the possible n tuples $(v_1, v_2, \ldots, v_n) \in GF(2^n)$ as input and produce only one of the two output bits $\{0, 1\} \in GF(2)$ [9, 48].

For example, the ith coordinate function $f(v_1, v_2, \ldots, v_n) = v_i$ is a Boolean function.

2.1.2.1 Representation of Boolean functions

A Boolean function $f : GF(2^n) \to GF(2)$ can be uniquely written in two different forms.

1. **Truth Table, TT**

   A truth table represent the possible outcome of a boolean function in a tabular form in which the first two columns correspond to possible inputs and the last column present the operation being performed.

   Example 2.1.7 Consider boolean function f is XOR function of two variables x_1 and x_2, represented as:

x_1	x_2	XOR
1	1	0
1	0	1
0	1	1
0	0	0

   TABLE 2.3: Truth Table of XOR Boolean function

   We can view a boolean function f as a binary vector f of size $(2^n \times 1)$, with the entries $f(x)$ indexed by the vectors $x \in GF(2^n)$.

   For example, the above boolean function can be represented as $f = [0\ 1\ 1\ 0]^\top$.

2. **Algebraic Normal Form (ANF)**

   The most used representations in cryptography is ANF of a boolean function. An ANF of a boolean function $f : GF(2^n) \to GF(2)$ is a polynomial of the following form: [47]

$$\begin{aligned} f(x_1, x_2, \ldots x_n) = a_0 \oplus & \\ & a_1 x_1 \oplus a_2 x_2 \oplus \ldots \oplus a_n x_n \oplus \\ & a_{1,2} x_1 x_2 \oplus \ldots \oplus a_{n-1,n} x_{n-1} x_n \oplus \\ & \ldots \\ & a_{1,2,\ldots n} x_1 x_2 \ldots x_n \end{aligned}$$

where $a_0, a_1, \ldots a_{1,2,\ldots n} \in \{0,1\}^n$. It was conventionally studied by researchers for the analysis of boolean functions. This ANF plays an important part in the analysis of boolean functions and S-boxes.

Example 2.1.8 Consider the "logical OR" boolean function represented as:

x_1	x_2	$x_1 \vee x_2$
1	1	1
1	0	1
0	1	1
0	0	0

TABLE 2.4: Truth Table of logical OR function

The ANF of "logical OR" boolean function is represented as:

$$f(x_1, x_2) = x_1 \oplus x_2 \oplus x_1 x_2$$

How many boolean functions can be defined for certain number of variables?

For one variable 4 boolean function can be defined as

x(variable)	false	x	not x	true
0	0	0	1	1
1	0	1	0	1

TABLE 2.5: One Variable Boolean Functions

The total number of boolean functions can be counted by formula 2^{2^n} where n is the number of variables. So for $n = 1$ we get $2^{2^1} = 4$.

For two variables as $n = 2$ we get $2^{2^2} = 16$ boolean functions and so on.

Number of Variables	Number of Boolean functions
0	$2^{2^0} = 2^1 = 2$
1	$2^{2^1} = 2^2 = 4$
2	$2^{2^2} = 2^4 = 16$
3	$2^{2^3} = 2^8 = 256$
4	$2^{2^4} = 2^{16} = 65536$
n	2^{2^n}

TABLE 2.6: Number of Boolean Functions

2.1.3 Properties of Boolean Functions

In cryptography, boolean functions are considered as a vital part for designing the substitution boxes so it is very important to study and choose them wisely satisfying certain crytographic properties for making cryptanalysis difficult for adversaries [38].

Definition 2.1.9 (Linearity of a Boolean Function)
A boolean function $f : GF(2^n) \longrightarrow GF(2)$ is considered to be linear if and only if it can be written in the form of linear combinations defined as

$$f(x_1, x_2, ..., x_n) = c_1x_1 \oplus c_2x_2 \oplus .....c_nx_n$$

where $\oplus$ is the XOR operation [12]. The linear combination of two boolean functions $f(x), g(x)$ is defined as

$$(f \oplus g)x = f(x) \oplus g(x)$$

Among the 2^{2^n} boolean functions on n variables, there are exactly 2^n linear functions.

Definition 2.1.10 (Affine Function)
A boolean function $f : GF(2^n) \longrightarrow GF(2)$ composed of a linear function and a constant is called an Affine function [48, 58]. It can be expressed as

$$f(x_1, x_2, ..., x_n) = c_1x_1 \oplus c_2x_2 \oplus .....c_nx_n \oplus c_0$$

Affine Cipher uses a boolean function over modulo M. It is a simple substitution cipher which can be broken easily because of its less security. This cipher performs addition and multiplication using the function given below

$$f(x) = (Ax \oplus C) \mod M$$

where A and C constitute the key used for encryption. For an input, the key will be applied and then we take the modulus M. For the encryption, we assign the following conversion of English alphabets to numbers.

A	0	*N*	13
B	1	*O*	14
C	2	*P*	15
D	3	*Q*	16
E	4	*R*	17
F	5	*S*	18
G	6	*T*	19
H	7	*U*	20
I	8	*V*	21
J	9	*W*	22
K	10	*X*	23
L	11	*Y*	24
M	12	*Z*	25

TABLE 2.7: Conversion in affine cipher

Consider the following example.

Example 2.1.11 Let the encryption function is

$$f(V) = (5V \oplus 2) \mod 26$$

suppose the plaintext message is "LEO" then

$$L = f(11) = 5 \mod 26$$
$$E = f(4) = 22 \mod 26$$
$$O = f(14) = 20 \mod 26$$

The ciphertext message is "FWU".

The decryption function is

$$V = [f(V) - 2] * 5^{-1} \mod 26$$
$$5^{-1} = -5 \mod 26$$
$$F = -5 * [5 - 2] \mod 26 = 11 = L$$
$$W = -5 * [22 - 2] \mod 26 = 4 = E$$
$$U = -5 * [20 - 2] \mod 26 = 14 = O$$

So we get the plaintext message "LEO".

Definition 2.1.12 (Hamming Weight and Hamming Distance)
The number of non-zero digits in a binary sequence is called its Hamming weight. It is denoted by $H(w)$ or Hwt or wt, where $w \in GF(2^n)$.
For example

$$w = 111001 \text{ then } H(111001) = 4.$$

Hamming distance between two functions

$$f(v), g(v) : GF(2^n) \longrightarrow GF(2)$$

is defined as [55]:

$$d(f, g) = H(f(v) \oplus g(v))$$

Here,

$$f(v) \oplus g(v) = f(v_0) \oplus g(v_0) \oplus f(v_1) \oplus g(v_1) \oplus \ldots \oplus f(v_{2^n-1}) \oplus g(v_{2^n-1})$$

where $v = (v_0, v_1, \ldots, v_{2^n-1}) \in GF(2^n)$
It is considered as the number of inputs where the functions differ or how many bits need to be changed in truth table of f to get g [58].

Example 2.1.13 Consider two Boolean functions

$$f(x) = 0 \;\; 1 \;\; 0 \;\; 1 \;\; 0 \;\; 1 \;\; 1 \;\; 1$$
$$g(x) = 0 \;\; 1 \;\; 1 \;\; 0 \;\; 1 \;\; 1 \;\; 1 \;\; 0$$
$$d(f, g) = 4$$

Example 2.1.14 Consider two Boolean functions

$$f(v) = v_1v_2v_3 \text{ and } g(v) = v_1 \oplus v_2 \oplus v_3$$

with input bits v_1, v_2, v_3. Hamming distance of these boolean functions is

$$d(f,g) = H(f(v) \oplus g(v))$$
$$= H(v_1v_2v_3 \oplus v_1 \oplus v_2 \oplus v_3) \ .$$

i	$v_i = v_1v_2v_3$	$(f \oplus g)(v_i)$
0	0 0 0	0
1	0 0 1	1
2	0 1 0	1
3	0 1 1	0
4	1 0 0	1
5	1 0 1	0
6	1 1 0	0
7	1 1 1	0

TABLE 2.8: Hamming distance between two functions f and g

Thus Hamming distance of f and g is 3.

Definition 2.1.15 (Bijection)

Bijection is a mapping in which each input bit mapped to exactly one and unique output bit making it a one-one and onto correspondence.

For a boolean function $f : GF(2^n) \longrightarrow GF(2)$, n be the possible input bits such as $\{0,1\}^n$ there exist a unique output bit, every output vector should appear once.

Definition 2.1.16 (Walsh Hadamard Transform)

A measure of similarity or correlation between the boolean function

$$f : GF(2^n) \longrightarrow GF(2)$$

and all linear functions $a \cdot x$ or linear combinations is known as Walsh Hadamard Transform of f, denoted by WHT_f [8]. It is defined as

$$WHT_f(a) = \sum (-1)^{f(x) \oplus a \cdot x} \qquad \forall \quad x, a \in GF(2^n)$$

where $a \cdot x = \sum_i a_i x_i$ is the inner product of vectors a and x.

The Walsh Hadamard transform is also called Walsh transform or Walsh Fourier transform.

We explain further that we calculate this similarity of boolean function by considering Hamming distances. Hamming distance or expected distance is calculated

by counting the bit positions which are different in the truth table of two boolean functions while the unexpected distance is the amount of change by which this distance differs from our expectation.

The expected distance of a boolean function $f : GF(2^n) \longrightarrow GF(2)$ with an affine function a_n is defined as

$$ED(f) = \frac{2^n}{2}$$

and the difference from this value will be considered as unexpected distance.

Example 2.1.17 Consider all possible affine functions and all boolean functions $f : GF(2^3) \longrightarrow GF(2)$, then we construct a truth table for affine functions.

Consider a boolean function $f = [1\ 0\ 0\ 1\ 1\ 0\ 0\ 1]$.

Its expected distances will be calculated with respect to all affine functions in truth table.

By above formula

$$ED(f) = \frac{2^n}{2} = \frac{2^3}{2} = 4$$

We calculate unexpected distances considering the difference between Hamming distance and $ED(f)$. Calculated values are presented in the following table:

Affine functions	truth table	Hamming distance	Unexpected distance
1	1 1 1 1 1 1 1 1	4	0
x_0	0 1 0 1 0 1 0 1	4	0
x_1	0 0 1 1 0 0 1 1	4	0
$x_1 + x_0$	0 1 1 0 0 1 1 0	8	4
x_2	0 0 0 0 1 1 1 1	4	0
$x_2 + x_0$	0 1 0 1 1 0 1 0	4	0
$x_2 + x_1$	0 0 1 1 1 1 0 0	4	0
$x_2 + x_1 + x_0$	0 1 1 0 1 0 0 1	4	0

TABLE 2.9: Calculation of Hamming distances and Unexpected distances of $f = [1\ 0\ 0\ 1\ 1\ 0\ 0\ 1]$

The Walsh transform of f is the maximum absolute value of all unexpected distances. $WHT_f = 4$.

There is a close relation between Walsh transform and Walsh Hadamard Matrix and both are used for cryptographical analysis of S-boxes.

Definition 2.1.18 (Walsh Hadamard Matrix)

"Walsh Hadamard matrix is defined as a $(n \times n)$ matrix, for n being a particular natural number, with all entries $+1$ or -1 such that its all rows and columns are orthogonal i.e their dot product is zero. It means that every two different rows have matching entries in exactly half of their columns and mismatched entries in the remaining columns. It is a consequence of this definition that the corresponding properties hold for columns as well as rows" [23].

The Hadamard matrices of dimension 2^n are given by the recursive formula.

The lowest order of Hadamard matrix is 2.

$$H(2^1) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H(2^2) = H(4) = \begin{bmatrix} H(2) & H(2) \\ H(2) & -H(2) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

and in general we get

$$H(2^n) = \begin{bmatrix} H(2^{n-1}) & H(2^{n-1}) \\ H(2^{n-1}) & -H(2^{n-1}) \end{bmatrix}$$

The coefficients calculated on the basis of Walsh Hadamard matrix are used for WHT_f.

We explain this with an example.

Example 2.1.19 Consider Walsh Hadamard Matrix of order 8.

$$H(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

We define a mapping $T : \{1, -1\} \longrightarrow \{0, 1\}$ from Hadamard matrix to 3-variable boolean functions. Results of this mapping are shown in the form of a truth table with the truth table of affine functions.

TT of Walsh Hadamard matrix mapping	TT of Affine functions
0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1
0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1
0 0 1 1 0 0 1 1	0 0 1 1 0 0 1 1
0 1 1 0 0 1 1 0	0 1 1 0 0 1 1 0
0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1
0 1 0 1 1 0 1 0	0 1 0 1 1 0 1 0
0 0 1 1 1 1 0 0	0 0 1 1 1 1 0 0
0 1 1 0 1 0 0 1	0 1 1 0 1 0 0 1

TABLE 2.10: Comparison of truth tables

This table shows that all the values are same except the first entry of table which is assigned because of constant. Thus Walsh Hadamard matrix generate the same results as by Walsh transform.

Example 2.1.20 Consider the Walsh transform calculated in Example 2.1.17 for $f = [1\ 0\ 0\ 1\ 1\ 0\ 0\ 1]$. With Walsh Hadamard matrix we get:

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The first value 4 will be ignored as it is the Hamming weight of function.
Thus $WHT_f = 4$. We can see the result is same as calculated by Walsh transform.

2.2 Substitution Boxes

Consider a function $F : \mathbb{V}_2^n \longrightarrow \mathbb{V}_2^m$ for some positive integers n and m where $\mathbb{V}_2$ is the finite field with two elements. Such functions F with given boolean functions $f_1, f_2, ...f_m$ are defined as

$$F(v) = (f_1(v), f_2(v), ...f_m(v))$$

at every $v \in \mathbb{V}_2^n$, called the coordinate functions of F. Such $(n \times m)$ functions are called vectorial boolean functions or S-boxes.

Thus a function $S : GF(2^n) \longrightarrow GF(2^m)$ which takes n-bits as input to produce m-bits as output is called an $(n \times m)$ S-box, defined as

$$v = S(u) = (f_1(u), f_2(u), \ldots, f_m(u)) \in GF(2^m) \qquad \forall u \in GF(2^n)$$

where f_i corresponds m-variable boolean functions.

Example 2.2.1 For example consider a 4×4 S-box that takes 4 input bits to produce 4 output bits. The first column of table represent input for S-box which is the Galois field $GF(2^4)$ and S-box is given in the second column.

GF(2^4)	S-box	f_1	f_2	f_3	f_4
0	9	1	0	0	1
1	13	1	0	1	1
2	10	0	1	0	1
3	15	1	1	1	1
4	11	1	1	0	1
5	14	0	1	1	1
6	7	1	1	1	0
7	3	1	1	0	0
8	12	0	0	1	1
9	8	0	0	0	1
10	6	0	1	1	0
11	2	0	1	0	0
12	4	0	0	1	0
13	1	1	0	0	0
14	0	0	0	0	0
15	5	1	0	1	0

TABLE 2.11: (4×4) S-box

In this table S-box entries are shown in binary format, where each column represent a boolean function f_i for $1 \leq i \leq 4$ of S-box. S-box properties depend on the all boolean functions which are used for constructing it.

Different S-boxes can be considered for the checking of MATLAB code performance but for analysis and confirmation of results produced by this MATLAB tool, we consider the AES S-box [42] as our main example. In fact, this is probably the most widely studied cipher at present for improving the strength of S-boxes to enhance the security parameters. Before we delve deeply into the properties of S-boxes, consider some of the mathematics in constructing the S-box design.

2.2.1 The Rijndael S-box (AES)

In 2001 Vincent Regimen and John Daemon gave a more complicated algorithm called Rinjindael, which was named as Advanced Encryption Standard [42]. It was a symmetric block cipher and used a key of 128 | 192 | 256 bits to encrypt 128 bit data, having a block of 16 bytes. The purpose of this S-box was to gain the non-linearity.

AES S-box was taken as, $m = n = 8$ and all operations were performed in the Galois field $GF(2^8)$. With this S-box one byte was replaced by another byte by various rounds of AES encryption algorithm.

2.2.1.1 Irreducible Polynomial

In AES, S-box was constructed with a specific irreducible polynomial in Galois field $GF(2^8)$ which is $x^8 + x^4 + x^3 + x + 1$
In hexadecimal, this is $11B$ and in binary, it is 100011011
This is important to remember that any irreducible polynomial can be used for generating S-boxes but why we used an irreducible polynomial instead of any polynomial, the reason is that Rijndael was creating basically diffusion with a nonlinear permutation function.
Why only this irreducible polynomial was chosen?.
Can we say it has some special characteristics that can make it more worthy for cryptography?.
For getting answers of these questions, consider the words of the developer of AES who states that while considering the multiplication in $GF(2^8)$, this polynomial $m(x) = 11B$ was by chance the first one in the list of irreducible polynomials of degree 8. So it means at that particular time it was just a chance to consider this polynomial but it was proved to be a very good decision later with further study in AES with other irreducible polynomials.

2.2.1.2 Affine Transformation

This concept was originally introduced in graphics for transformation in graphs. In general, an affine transform is composed of linear transformations (rotation, scaling) and a translation (or shift). As we can see that moving pixels in different directions is quite similar to moving entry values at different locations in a matrix, so this concept was applied to matrices in AES.

For the AES $(0 \times 1F)$ is the affine matrix, the whole process can be shown in matrix form as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix}$$

This affine transformation basically shows multiple rotations of the byte and addition is the XOR operation.

Example 2.2.2 For an input 0×53 in AES, we first find its inverse, which is $0 \times CA$.

"The input bits are multiplied with the bits of a given row and so on. Finally, all bits are XORed against each other within that row and we get transformed bit for

that row [30].

$$\begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix} = \begin{bmatrix} 1 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 0 \oplus 0 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 1 \oplus 1 \\ 1 \cdot 0 \oplus 1 \cdot 1 \oplus 0 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 0 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 1 \oplus 1 \\ 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 0 \cdot 0 \oplus 0 \cdot 0 \oplus 0 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 1 \oplus 0 \\ 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 0 \cdot 0 \oplus 0 \cdot 0 \oplus 0 \cdot 1 \oplus 1 \cdot 1 \oplus 0 \\ 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 0 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 1 \oplus 0 \\ 0 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 1 \oplus 1 \\ 0 \cdot 0 \oplus 0 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 0 \cdot 1 \oplus 1 \\ 0 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 1 \oplus 0 \end{bmatrix}$$

$$\begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Studies showed that S-boxes generated by using affine transformations gave maximum non-linearity." Further work in this field was proposed by [5, 6, 14, 35].

2.2.2 Properties of Cryptographically Strong S-boxes

Substitution boxes are a crucial part in the science of cryptography so we study some desirable properties for a cryptographically strong S-box.

1. **Balanced**

   A boolean function $S : GF(2^n) \longrightarrow GF(2)$ is called balanced if the output set contains equal number of ones and zeros in the corresponding truth table.

   Example 2.2.3 We provide a comparison of balanced and unbalanced functions. Consider two boolean functions, XOR and AND defined as:

   $$S_1 = \oplus : GF(2^2) \longrightarrow GF(2)$$
   $$S_2 = \cdot : GF(2^2) \longrightarrow GF(2)$$

   They define following truth table for two variables x_1 and x_2.

x_1	x_2	$x_1 \oplus x_2$	$x_1 \cdot x_2$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

   TABLE 2.12: Truth Table of XOR, AND functions

   Third column has equal number of zeros and ones representing "XOR" function which is balanced while fourth column presents "AND" function which is not balanced.

2. **Bijective**

   A boolean function $S : GF(2^n) \longrightarrow GF(2)$ is bijective if and only if all linaer combinations of columns are balanced. To check the bijective property of an $(n \times n)$ S-box a method was introduced in [50] which states that "The

bijective property is satisfied if for the boolean functions f_i (for $1 \leq i \leq n$) of S-box, this condition holds

$$Hwt(\sum_{i=1}^{n} c_i f_i) = 2^{n-1} \tag{2.1}$$

where $c_i \in \{0,1\}$ for $(c_1, c_2, \ldots, c_n) \neq (0, 0, \ldots, 0)$ and Hwt is the Hamming weight" [56].

Condition (2.1) in fact, guarantees that every boolean function f_i and all their combinations are balanced.

Consider the S-box presented in Table 2.11. Note that all boolean functions have $Hwt(f_i) = 8$ for $1 \leq i \leq 4$, so f_i are balanced. Thus the given S-box is balanced and bijective.

3. **Non-linearity**

The *Non-linearity*, $NL(f)$, of any boolean function

$$f : GF(2^n) \longrightarrow GF(2)$$

is defined as the number of bits which must be changed in the truth table to get the closest affine function [40].

We measure non-linearity of a boolean functions using Walsh transform or Walsh Hadamard matrices. Distances to linear functions can be computed easily which are exactly the same as computed by Walsh transform which we have discussed in Example 2.1.17 .

For an S-box $S : GF(2^n) \longrightarrow GF(2^m)$ such that

$$S(u) = v \qquad \text{for} \qquad v \in GF(2^m); u \in GF(2^n)$$

non-linearity can be calculated as: [10]

$$NL_S = 2^{n-1} - \frac{1}{2} \max |W(u, v)|$$

where $W(u, v)$ is Walsh transform defined as:

$$W(u, v) = \sum_{x \in GF(2^n)} (-1)^{v \cdot f(x) \oplus u \cdot x}$$

Here we use notation $W(u, v)$ intead of WHT_f since it is defined for a vectorial boolean function S. Walsh Spectrum can be defined as:

$$WS = \{W(u, v) : u \in GF(2^n), v \in GF(2^m)\}$$

Non-linearity is the minimum Hamming distance between all boolean functions $v \cdot f$ and all affine functions $u \cdot x$ where "$\cdot$" is the inner product in respective Galois field.

Example 2.2.4 Consider S-box $S : GF(2^4) \longrightarrow GF(2)$ in Table 2.11. We calculate Walsh transform for one entry $(0, 0, 0, 0)$ of S-box as:

$$W((0,0,0,0),(1,0,0,1)) = \sum_{x \in GF(2^4)} (-1)^{(1,0,0,1) \ \cdot \ f(x) \ \oplus \ (0,0,0,0) \ \cdot \ x}$$

By considering all entries of S-box, we calculate

$$\begin{aligned} W((0,0,0,0),(1,0,0,1)) = {} & (-1)^{(1,0,0,1)\cdot(1,0,0,1) \ \oplus \ (0,0,0,0)\cdot(0,0,0,0)} \\ & + (-1)^{(1,1,0,1)\cdot(1,0,0,1) \ \oplus \ (0,0,0,1)\cdot(0,0,0,0)} \\ & + (-1)^{(1,0,1,0)\cdot(1,0,0,1) \ \oplus \ (0,0,1,0)\cdot(0,0,0,0)} \\ & + \ldots \\ & + (-1)^{(0,0,0,0)\cdot(1,0,0,1) \ \oplus \ (1,1,1,0)\cdot(0,0,0,0)} \\ & + (-1)^{(0,1,0,1)\cdot(1,0,0,1) \ \oplus \ (1,1,1,1)\cdot(0,0,0,0)} \end{aligned}$$

Similarly we can calculate the Walsh transform value for each entry of an S-box and calculate non-linearity by defined formula.

If we know the non-linearities of all boolean functions which are constructing S-Box, the minimum of these will be the non-linearity of that S-box.

We can calculate non-linearity by Walsh Hadamard matrix also.

Example 2.2.5 Consider a boolean function $S : GF(2^4) \longrightarrow GF(2)$. For one boolean function of S-box, let

$$f = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

8×8 Hadamard Matrix is

$$H(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

We then calculate Wlash spectrum value by multiplying f and $H(8)$ and get

$$f \times H(8) = \begin{bmatrix} 4 & 2 & -2 & 0 & 0 & -2 & -2 & 0 \end{bmatrix}$$

Minimum absolute value is 2. So we get

$$NL_f = 2^{3-1} - \mid 2 \mid = 2$$

We will not consider the first element 4 because this is the Hamming weight of boolean function f.

The maximum non-linearity in $GF(2^n)$ is

(a) for $n = even$ is $2^{n-1} - 2^{\frac{n}{2}-1}$

(b) for $n = odd$ is $2^{n-1} - 2^{\frac{n-1}{2}}$

A boolean function f is called perfect nonlinear if the function

$$f(x \oplus a) \oplus f(x)$$

is balanced for all $a \in GF(2^n)$ where $a \neq 0$.

4. **Bent Functions**

   S-Boxes are required to be composed of highly nonlinear boolean functions. We can define different boolean functions with great non-linearity value but bent functions are basically a special kind of boolean functions which have maximum non-linearity.

   Bent functions are defined by Walsh transform as

$$\hat{f}(s) = \frac{1}{\sqrt{2^n}} \sum_{u \in GF(2^n)} (-1)^{f(u) \oplus s.u}$$

   Thus a boolean function f which can attains maximum non-linearity is called Bent function [38].

   Example 2.2.6 By maximum non-linearity criterion, we know that in $GF(2^2)$, the function with non-linearity 1 is bent.

$$NL(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$$
$$2^{2-1} - 2^{\frac{2}{2}-1} = 2 - 1 = 1$$

   Clearly bent functions are not linear and not affine but bent functions are a special type of boolean function which have highest Strict Avalanche Criterion (SAC) [24] and Bit Independence Criterion (BIC) [53] which will be explained later.

   MATLAB code (SAMT) which we present here will calculates maximum non-linearity for given Galois field and compare it with the calculated non-linearity of given S-box so user can check also that how many boolean functions are bent in the given S-box.

5. **XOR Table**

   For an S-box $S : GF(2^n) \longrightarrow GF(2^m)$, we consider two elements

   $$u \in GF(2^n) \quad \text{and} \quad v \in GF(2^m)$$

   then the XOR table entry corresponding to (u, v) is given by [33]:

   $$XOR(u, v) = \# \ \{ \ x \in GF(2^n) \ : \ S(x) \oplus S(x \oplus u) = v \ \}$$

   where "#" denotes the cardinality of set.[31].

   Example 2.2.7 We build every possible pair (p_1, p_2) that can be given to the S-box. For example consider the AES S-box and take $(00, 00)$ and calculate the XOR of every pair $(p_1 \oplus p_2) \in GF(2^n)$ which will be 00.

   Now look the transformed values in S-box $(c_1 = S(p_1), c_2 = S(p_2))$ which is $(63, 63)$ and calculate the $(c_1 \oplus c_2) \in GF(2^m)$ which gives (00).

   Then increment this value in the row $(p_1 \oplus p_2)$ and the column $(c_1 \oplus c_2)$ to reflect basically that this input difference lead to this output difference.

   This table would be too complex as for AES it would be a 256×256 table.

   In XOR table, all entries should be 0 or 2 which is theoretically the best possibility. All the values are even and they should be as small as possible for having good resistance against differential attacks. The XOR value for an S-Box is the highest XOR table entry.

6. **Dynamic Distance**

   "The Dynamic Distance (DD) of order j for a boolean function [22]

   $$f : GF(2^n) \longrightarrow GF(2)$$

   is defined as:

   $$DD_j(f) = \max_{1 \leq wt(d) \leq j} \ \frac{1}{2} \left| 2^{n-1} - \sum_{x \in GF(2^n)} f(x) \oplus f(x \oplus d) \right|$$

where $d \in \{0,1\}^n$.

It provides a measure for other dynamic properties such as SAC which will be satisfied if DD has small integral value and closer to zero".

For calculating Dynamic distances DD_1, we consider specific matrix $d \in GF(2^n)$ whose each entry should have hamming weight 1.

Example 2.2.8 Consider the example of S-box presented in Table 2.11 in $GF(2^4) \longrightarrow GF(2^4)$ and matrix d as

$$d = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Calculated dynamic distances of boolean functions are [4 4 4 2].

For AES S-box, $S : GF(2^8) \longrightarrow GF(2^8)$, this matrix d is defined as

$$d = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

For $S(u) = v$, first calculate $u \oplus d$ for each entry of all boolean functions with every entry of this matrix d then we calculate

$$f(u) \oplus f(u \oplus d)$$

It will finally contribute to calculate the DD for S-box.

7. **Avalanche Effect**

   If by changing a single input bit, half of the output bits changed then this is called Avalanche effect. [24]

   Consider a boolean function $f : GF(2^n) \longrightarrow GF(2)$ and choose a pair of n-bit plaintext vectors X and X_j which is dissimilar to X only in jth bit. Their corresponding output bits are $f(X)$ and $f(X_j)$ which are different at least in i bit. Taking XOR of output bits, we get:

   $$V_j = f(X) \oplus f(X_j)$$

   Each V_j containing n-bits are called avalanche variables. If the above procedure is repeated for $1 \leq j \leq n$ times and for each j, half of the variables are equal to 1, then we said that f has a good avalanche effect.

8. **Strict Avalanche Criterion**

   This criterion states that by changing one input bit, each output bit will change with probability of one half. To satisfy the criterion basically the boolean function should has a 50% dependency on each of its input bits.

   SAC can be measured using the hamming weight of boolean function. Recall that the hamming weight of a specific binary vector is the occurrence of 1 in that vector. Therefore, if you have an input of 8 bits with three 1 and an output of four 1, the hamming weight of the output is 4. A boolean function satisfies the SAC or not, can be checked by measuring the hamming weights of the input and output values and then comparing them.

   For an S-box $S : GF(2^n) \longrightarrow GF(2^m)$ where $S(u) = v$,
   Difference Distribution vectors of each boolean function can be calculated as:

   $$SA_i(f) = \sum_{u \in GF(2^n)} f(u) \oplus f(u \oplus d_i)$$

   where d matrix is defined in the same way as for calculating Dynamic Distance.

A boolean function f satisfies the SAC if this condition holds

$$SA_i(f) = 2^{n-1}$$

If the value of dynamic distance DD is a small integer close to zero, S-box satisfies the SAC.

In this way presented MATLAB code will calculate the SAC for each boolean function of S-box and a Dependence matrix will be computed showing all these values for the user.

9. **Bit Independence Criterion**

The Bit Independence Criterion (BIC) is another desirable property for designing strong S-boxes. As we increase the independence of bits with each other, it gradually harder to guess the design of system. It also shows that all variables should be independent pairwise for a specific set of avalanche vectors which are produced by single plaintext bit transpose [53].

"To measure this criterion in an S-box $S : GF(2^n) \longrightarrow GF(2^m)$, the correlation coefficient is needed between the jth and kth components of the output string, which is called the avalanche vector A^{ei}.
A bit independence parameter corresponding to the effect of ith input bit change on the jth and kth bits of avalanche vector is defined as [52]

$$BIC(a_j, a_k) = \max_{1 \le i \le n} \mid corr(a_j^{ei}, a_k^{e\acute{i}}) \mid$$

BIC criterion for an S-box function f is

$$BIC(f) = \max_{1 \le j,k \le n} BIC(a_j, a_k) \quad for \;\; j \neq k$$

$BIC(f)$ takes the values in $[0, 1]$ and in the worst case it is equal to 0".

For any boolean function f, consider two output bits of an S-box $f(r)$ and $f(s)$ where $r \neq s$, the S-box satisfies the BIC, if $f(r) \oplus f(s)$ is a highly nonlinear function and come closer to satisfy the SAC [11].

10. **Differential Branch Number**

    Differential branch number of an S-box $S : GF(2^n) \longrightarrow GF(2^m)$ is calculated by the formula: [44]

$$BN = \min_{u_1, u_2 \neq u_1} [wt(u_1 \oplus u_2) + wt(S(u_1) \oplus S(u_2))]$$

    where $u_1, u_2 \in GF(2^n)$.

    MATLAB code presented in thesis calculates all possible XORs between the Galois field $GF(2^n)$ entries and then all possible XORs between the S-box entries. Finally calculate their hamming weights and compute the minimum value.

    For a bijective S-box the Differential branch number is at least 2.

    The avalanche effect of any S-box is directly proportional to the branch number so it should be maximized for attaining best cryptographic properties which enhance the resistance of system against differential attacks.

11. **Algebraic Degree**

    An algebraic degree is related with the non-linearity measure [2]. "For a boolean function $f : GF(2^n) \longrightarrow GF(2)$, it is defined as the number of variables in highest order term with non-zero coefficients and can be expressed as

$$deg(f) = n - 1$$

    Higher algebraic degree is considered better than the lower."

    "The algebraic degree of a function is largest number of inputs appearing in any product in its algebraic normal form (ANF).
    For example, $x_1 \oplus x_2$ has degree 1 so it in linear and $x_1 \oplus x_1x_2x_3$ has defree 3" [12].

    The Boolean functions whose algebraic degree do not exceed 1 are called the affine functions.

12. **Absolute Indicator and Sum of Square Indicator**

    Derivative of a boolean function $f : GF(2^n) \longrightarrow GF(2)$ with respect to a point b is defined as:

$$D_f(b) = f(x) \oplus f(x \oplus b)$$

    On the basis of above defined derivative, Auto-correlation (AC) of a boolean function f can be defined on all $b \in GF(2^n)$ as:

$$\Delta_f(b) = \sum(-1)^{f(x)\oplus f(x\oplus b)} \quad \text{where} \quad x \in GF(2^n)$$

    The absolute indicator of boolean function f is defined as the maximum absolute value of AC except the origin, which can be expressed as [51]

$$\Delta_f = \max_{b\in GF(2^n), b\neq 0} \mid \Delta_f(b) \mid$$

    The Sum of square indicator [51] of boolean function f also derived from AC and can be expressed as

$$\sigma_f = \sum_{b\in GF(2^n)} (\Delta_f(b))^2$$

13. **Fixed and Opposite Fixed Points**

    Consider an S-box $S : GF(2^n) \longrightarrow GF(2^m)$ and for $u \in GF(2^n)$

    A point is called fixed point of S-box if

$$S(u) = u$$

    A point is called opposite fixed points of S-box if

$$S(u) = u'$$

    where u' is complement of u.

Example 2.2.9 Consider a 2×2 S-box with 2 boolean functions

GF(2)	Binary format GF(2)	S-box	Binary format S-box
0	00	1	01
1	01	3	11
2	10	2	10
3	11	0	00

TABLE 2.13: S-box fixed points

In this example "2" is a fixed point of S-box.

Example 2.2.10 Consider now

GF(2)	Binary format GF(2)	S-box	Binary format S-box
0	00	1	01
1	01	2	10
2	10	3	11
3	11	0	00

TABLE 2.14: S-box opposite fixed points

In this example "1" is the opposite fixed point of S-box.

Any S-box without fixed and opposite fixed points is considered to be better against differential cryptanalysis attacks as compared to those who have fixed and opposite fixed points.

MATLAB code compares bit wise each input byte with every byte of the given S-box to check the fixed points and then compare the complement of each input byte for opposite fixed points in the same manner. It counts those points and show the results.

Chapter 3

MATLAB tool for the analysis of S-boxes

The focus of this thesis is to develop a tool for calculating some desired cryptographic properties of S-boxes in any Substitution-Permutation Network. An S-box is utilized to provide the only nonlinear part of the cryptosystem so its strength enhances the security of whole system.

User can request for this MATLAB tool file on following emails.

rashid.ali@cust.edu.pk
fatimaishfaq88@gmail.com

3.1 MATLAB tool

This tool is designed in MATLAB software for not only the S-box but also for the boolean functions which are basically constructing that S-box. Any S-box defined as $S : GF(2^n) \longrightarrow GF(2^n)$ for $2 \leq n \leq 20$ can be checked with this tool so it really provides a large space regarding Galois fields for the analysis.

The hardest challenge for designing this tool is the choice of certain properties which we consider important for the analysis of an S-box. We try to cover some

properties which play a critical role in linear and differential cryptanalysis. As this MATLAB code compute values for the all boolean functions as well, so we consider the non-linearity SAC and BIC as the most important paramenters of an S-box. Thus user can have an in depth investigation of all these properties contribution regarding each boolean function of S-box.

Results of cryptographic properties of an S-box which are evaluated by this tool are briefly described below.

Fixed points of S-box Number of fixed points in S-box will be shown.

Opposite Fixed points of S-Box Number of opposite fixed points will be shown.

Bijective Result will be shown as S-box is bijective or S-box is not bijective.

Hamming weight of all boolean functions Hamming weights of all boolean functions will be shown in a row.

Balanced Result will be shown as S-box is balanced or S-box is not balanced.

Non-linearity of all boolean functions Non-linearity of all boolean functions will be shown in a row.

Average Non-linearity of all boolean functions Code will add non-linearities of all boolean functions and take the average.

Almost Bent Non-linearity value According to given Galois field, almost bent non-linearity value will be calculated.

Perfect Non-linearity value of boolean functions Perfect non-linearity of all boolean functions will be shown in a row.

Non-linearity of S-Box Non-linearity of S-box will be shown.

Maximum Non-linearity value for given Galois field Code will calculate the maximum non-linearity which can be achieved in given Galois field.

Differential Branch Number As a single output value, differential branch number for provided S-box will be shown.

Dynamic Distance Table First MATLAB code will generate all vectors in given Galois field with hamming weight 1. Dynamic distances of each boolean function will be calculated with every such vector. These results will be shown in the form of a matrix so user can check every boolean function contribution independently.

Dynamic Distances of Boolean functions Dynamic distance of a boolean function is the maximum of its all calculated values with every vector defined above. Thus dynamic distances of all boolean functions will be shown in the form of a row.

Desired Avalanche value for boolean function The desired avalanche effect value will be calculated for any given Galois field.

Avalanche value of Boolean functions Avalanche values of all boolean functions will be calculated and a matrix will be shown where each column gives calculated results for one boolean function.

Boolean functions results which satisfy Avalanche Criterion Total number of results from above matrix which satisfy avalanche criterion will be shown.

Avalanche Criterion Percentage of S-box According to the number of results satisfying avalanche criterion, percentage avalanche effect for complete S-box will be calculated.

Dependence Matrix of S-box Dependence matrix for given S-box will be shown represeting each boolean function as its column.

Desired SAC of a boolean function Desired value for a boolean function to satisfy SAC is $n \times 2^{n-1}$.

SAC values of boolean functions SAC results produced by every boolean function will be added and shown in a row.

Sum of Squares Indicator of boolean functions Sum of square indicator of all boolean functions will be presented in a row.

Absolute Indicator of boolean functions Absolute indicator of all boolean functions will be shown in a row.

BIC Non-linearity criterion of S-box A matrix will show BIC non-linearity of all boolean functions constituting S-box.

3.1.1 How to use this MATLAB code?

For new users of MATLAB software we refer them to MATLAB help

https://www.mathworks.com/help

which can provide a necessary guide for using the software efficiently. We are also providing here a brief detail for the new users so that they can use the tool without even extensive study of MATLAB.

3.1.1.1 Requirements

The basic requirements for using the MATLAB tool are:

1. A desktop computer or laptop with working windows 8 or Linux.
2. Installed MATLAB software with at least 2GB RAM.
3. Code file **S-box Analysis MATLAB Tool (SAMT.p)** saved in MATLAB directory. This particular file with extension "*.p*" can be run through MATLAB only. This file can also be saved in any folder where user has his already saved work of MATLAB and has pre-defined MATLAB path.

3.1.1.2 S-box Analysis MATLAB Tool (SAMT.p)

For using the SAMT.p file, user need to follow these steps.

1. First start MATLAB software and wait till it shows the command **"Ready"** in MATLAB window.

2. Now user can view two partitions showing "**Editor space**" and "**Command Window**" where user can see any code descripition in Editor space and can check the obtained results in Command Window.

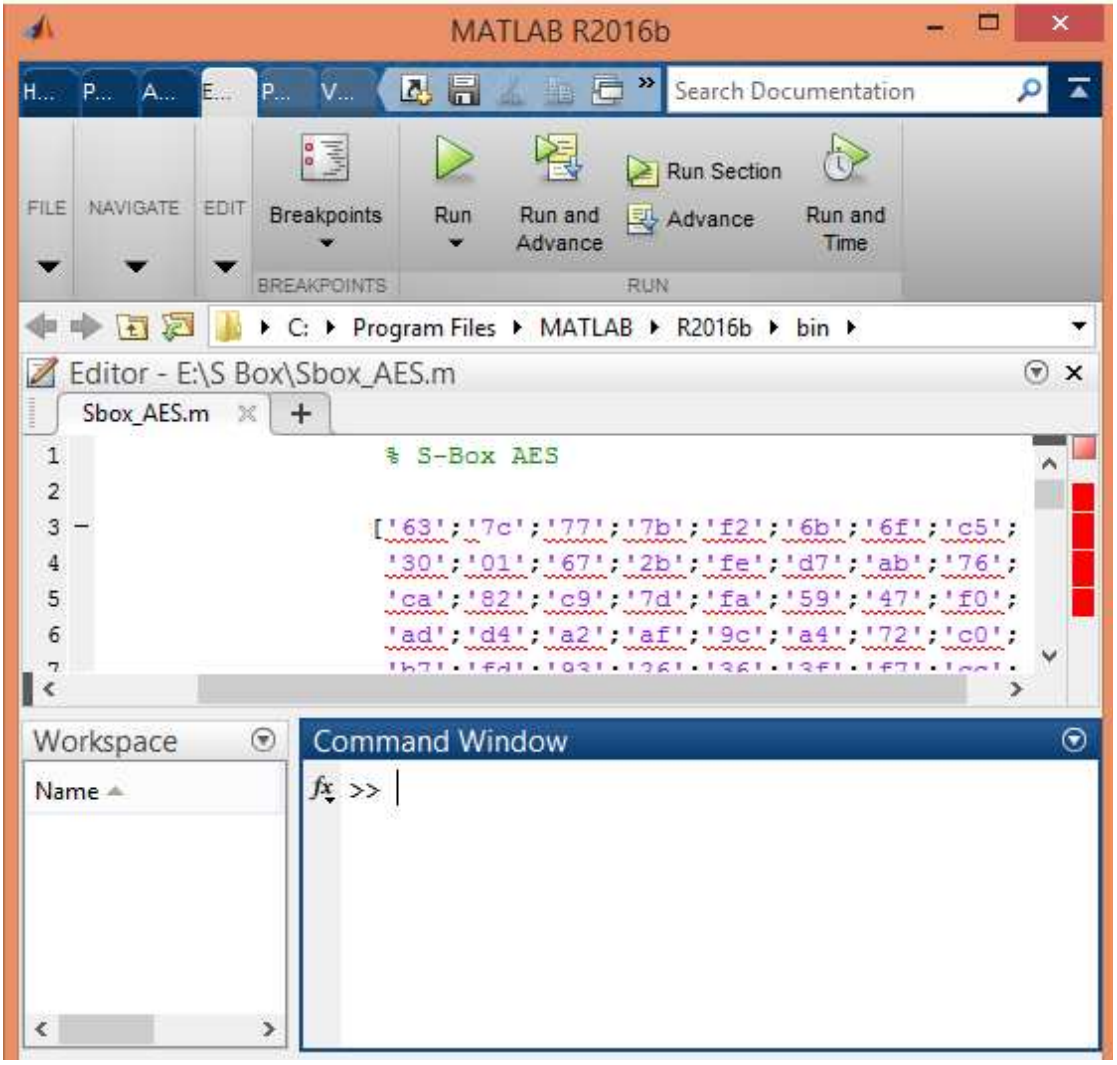

FIGURE 3.1: MATLAB Window with Editor space and Command window

3. Open particular folder where all MATLAB files are stored with code file SAMT.p

4. User will click on **"Run"** command for defining the MATLAB execution path.

5. A box will appear by MATLAB for confirming the path to start required calculation. User will click **"Add to Path"** for confirming the path. MATLAB then change its path to the specific directory of computer where user had saved the tool file.

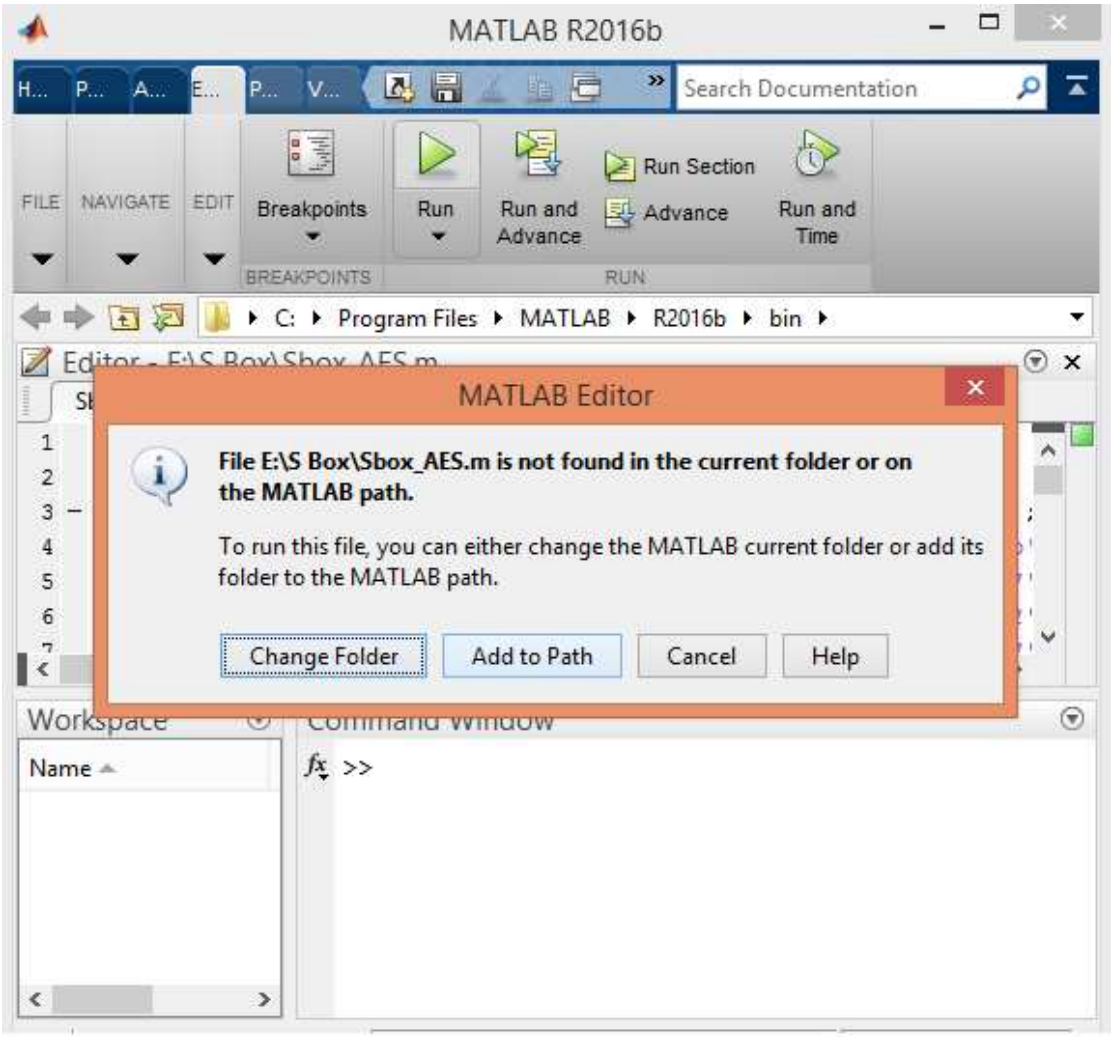

FIGURE 3.2: MATLAB Path requirement

6. Write code file name in Command Window without any extension and press enter.

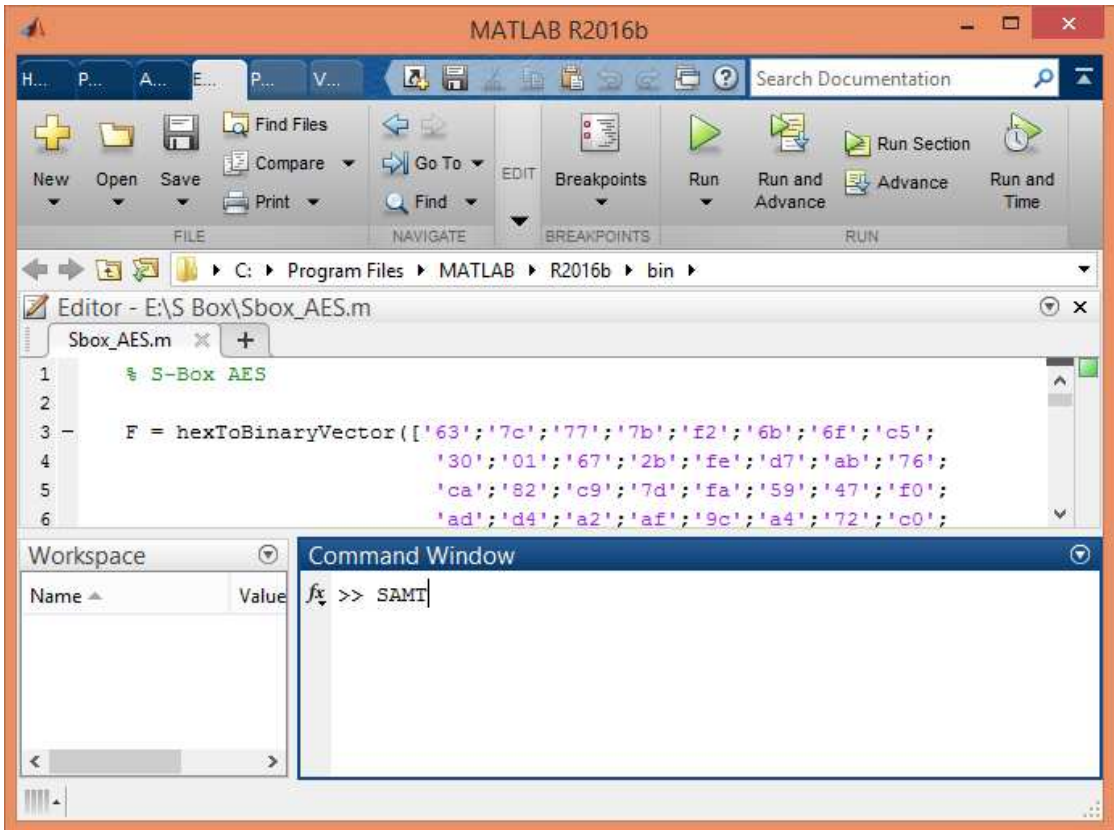

FIGURE 3.3: Code file name in Command Window

7. In Command Window, first command will appear as

   Type the value of n =

   Code requires the value of n for the specific Galois field $GF(2^n)$ in which user wants to enter the S-box. Thus user can type any value of n from 2 to 20 and press enter.

8. Second command will appear as

   Data Type (Hexadecimal = 1 and Decimal = 2) =

   User can input the S-box entries either in decimal or hexadecimal formats only. The values of S-box should be entered in decimal format for $2 \leq n \leq 7$ according to MATLAB input requirement. MATLAB code will convert them into binary numbers for its further execution process.

User will write "1" for entering the Hexadecimal values if he wants to use the $n \geq 8$ in Galois field $GF(2^n)$ and then press enter.

User will write "2" if entering the Decimal values for the $n \geq 2$ according to his S-box in Galois field $GF(2^n)$ and then press enter.

9. Third command will appear as

$$\text{Enter the values of S-box} =$$

User can either copy and then paste the values of S-box in this space from a text file where he had already written or stored the values or he can write the S-box entries directly here.

User need to input the values in a specific style because MATLAB software can read the Decimal and Hexadecimal values in a particular format which we explain here with examples so that user can understand these systems conversion in MATLAB.

For Hexadecimal format one entry, let a, has to be enclosed in commas with brackets ($'a'$).

Example 3.1.1 For example, enter the value in Command Window of MATLAB usinga bulit-in-command and press enter.

$$hexToBinaryVector\ (\ '7c'\ )$$

MATLAB will show the result in binary number: 1 1 1 1 1 0 0

or if user write this value and enter.

$$hexToBinaryVector\ (\ '7c'\ ,\ 8\ )$$

it will give the binary result with 8 bits as: 0 1 1 1 1 1 0 0

Now consider AES S-box for entering in Hexadecimal format, user will write each entry, let a, enclosed in commas $'a'$ separate them by ; and enclose by

brackets [ ] as follows: (but will not write the conversion "hexToBinaryVector")

$$[\,'63\,';\,'7c\,';\,'77\,';\;\ldots\;;\,'6f\,';\,'30\,';\,'01\,']$$

Code will accept 256 entries for AES S-box, convert them into binary format and start execution.

For Decimal format consider following examples.

Example 3.1.2 For entering a single entry in decimal format, write this in Command window using MATLAB built-in-command and enter

$$de2bi(\ 161\ )$$

MATLAB will show an 8 bit result as: 1 0 0 0 0 1 0 1

Now for entering a complete S-box in the decimal format, user will write them in commas, separated with space enclosed by brackets ( ) in the following way. (but will not write the conversion "de2bi")

$$(\,'161\ \ 85\ \ 129\ \ 224\ \ \ldots\ \ 207\ \ 177\ \ 48\ \ 205\ \ 68\ \ 60\ \ 1\,')$$

Code will accept 256 entries, convert them into binary format and start calculation.

10. When code completes its execution for given S-box, results will appear in the Command window mentioning all the properties.

11. Code will also provide bent value for a boolean function, maximum non-linearity value for given Galois field, desired avalanche effect for boolean functions and desired SAC value for a boolean function. Thus user can compare given S-box results with desired values in particular Galois field.

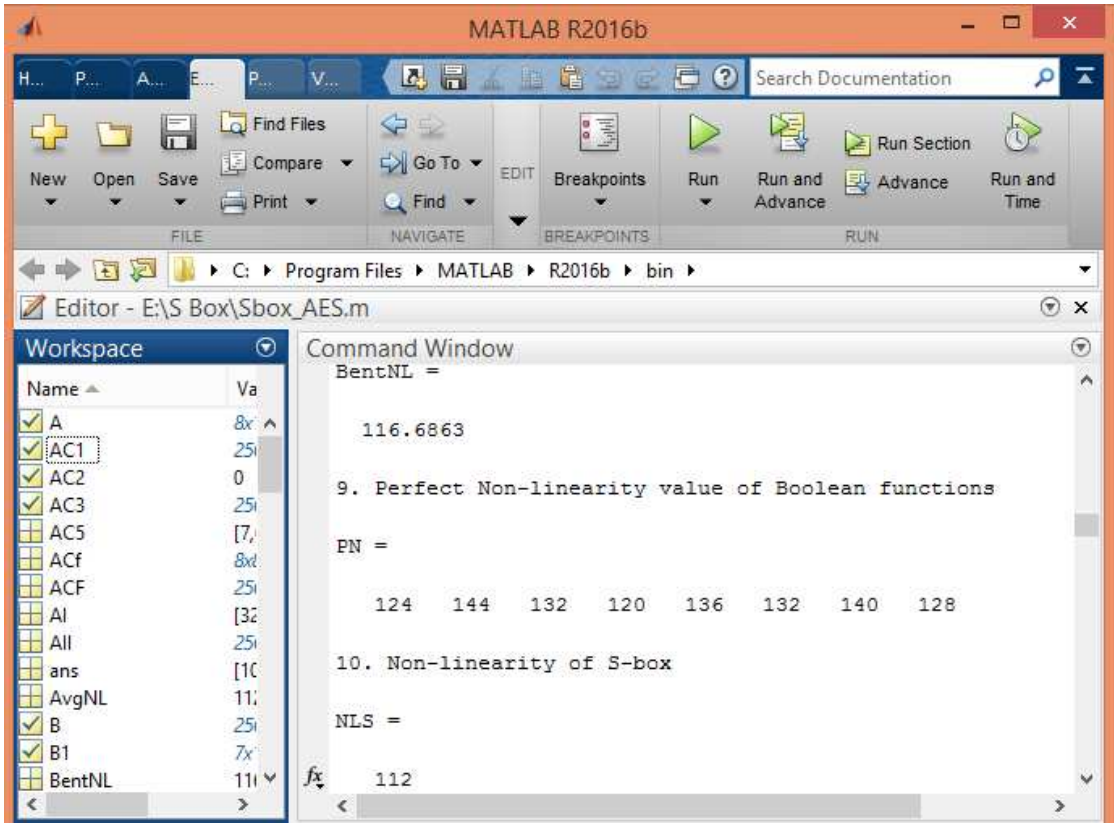

FIGURE 3.4: MATLAB code Results

12. As we consider all properties discussed above are important for the analysis of an S-box, so code will calculate all these properties.

13. MATLAB code execution process is fast enough as it calculates all result for AES S-box in less than 1 minute.

3.1.1.3 Notation

For programming, we consider S-box as a boolean function

$$S : GF(2^n) \longrightarrow GF(2^n)$$

As an input, a file that contains all elements of $GF(2^n)$ (according to given input of n) will be automatically generated by the code for considering the domain of function S. In the MATLAB code, following notations are used:

Z Set of elements $GF(2^n)$, representing domain of S-box.

F Set of elements of S-box.

fixp Fixed points of S-box.

opfixp Opposite fixed points of S-box.

WS Wlash spectrum of boolean functions.

WH Walsh Hadamard transform of boolean functions.

HW Hamming weight of boolean functions.

NLf Non-linearity of boolean functions.

AvgNL Average of non linearities of all boolean functions.

BentNL Bent non-linaerity value of a boolean function.

PN Perfect non-linearity value of boolean functions.

NLS Non-linearity of S-box.

MaxNL Maximum non-linearity value of boolean function.

DBN Differential branch number of S-box.

e The matrix containing all vectors with hamming weight 1.

DD Dynamic distance table

DDf Dynamic diatances of boolean functions.

ACf Avalanche effect value of S-box entries.

NfAC Number of entries in S-box which satisfy Avalanche criterion.

PAC Avalanche criterion percentage value of S-box.

DM Dependence matrix of S-box.

SAC Desired SAC value.

SACf SAC values of boolean functions.

SSI Sum of Squares Indicator of Boolean functions.

AI Absolute Indicator of Boolean functions

BICNL BIC Non-linearity criterion of S-box

3.2 Analysis of different S-boxes by SAMT

We analyze the properties of some S-boxes with SAMT and presented the obtained results.

3.2.1 AES S-box Analysis

AES [42] S-box represented in hexadecimal format for the MATLAB code will be written in the following way.

$$
\begin{aligned}
&\text{“}[\prime 63\prime;\prime\ 7c\prime;\prime\ 77\prime;\prime\ 7b\prime;\prime\ f2\prime;\prime\ 6b\prime;\prime\ 6f\prime;\prime\ c5\prime;\\
&\prime 30\prime;\prime\ 01\prime;\prime\ 67\prime;\prime\ 2b\prime;\prime\ fe\prime;\prime\ d7\prime;\prime\ ab\prime;\prime\ 76\prime;\\
&\prime ca\prime;\prime\ 82\prime;\prime\ c9\prime;\prime\ 7d\prime;\prime\ fa\prime;\prime\ 59\prime;\prime\ 47\prime;\prime\ f0\prime;\\
&\prime ad\prime;\prime\ d4\prime;\prime\ a2\prime;\prime\ af\prime;\prime\ 9c\prime;\prime\ a4\prime;\prime\ 72\prime;\prime\ c0\prime;\\
&\prime b7\prime;\prime\ fd\prime;\prime\ 93\prime;\prime\ 26\prime;\prime\ 36\prime;\prime\ 3f\prime;\prime\ f7\prime;\prime\ cc\prime;\\
&\prime 34\prime;\prime\ a5\prime;\prime\ e5\prime;\prime\ f1\prime;\prime\ 71\prime;\prime\ d8\prime;\prime\ 31\prime;\prime\ 15\prime;\\
&\prime 04\prime;\prime\ c7\prime;\prime\ 23\prime;\prime\ c3\prime;\prime\ 18\prime;\prime\ 96\prime;\prime\ 05\prime;\prime\ 9a\prime;\\
&\prime 07\prime;\prime\ 12\prime;\prime\ 80\prime;\prime\ e2\prime;\prime\ eb\prime;\prime\ 27\prime;\prime\ b2\prime;\prime\ 75\prime;\\
&\prime 09\prime;\prime\ 83\prime;\prime\ 2c\prime;\prime\ 1a\prime;\prime\ 1b\prime;\prime\ 6e\prime;\prime\ 5a\prime;\prime\ a0\prime;\\
&\prime 52\prime;\prime\ 3b\prime;\prime\ d6\prime;\prime\ b3\prime;\prime\ 29\prime;\prime\ e3\prime;\prime\ 2f\prime;\prime\ 84\prime;\\
&\prime 53\prime;\prime\ d1\prime;\prime\ 00\prime;\prime\ ed\prime;\prime\ 20\prime;\prime\ fc\prime;\prime\ b1\prime;\prime\ 5b\prime;\\
&\prime 6a\prime;\prime\ cb\prime;\prime\ be\prime;\prime\ 39\prime;\prime\ 4a\prime;\prime\ 4c\prime;\prime\ 58\prime;\prime\ cf\prime;\\
&\prime d0\prime;\prime\ ef\prime;\prime\ aa\prime;\prime\ fb\prime;\prime\ 43\prime;\prime\ 4d\prime;\prime\ 33\prime;\prime\ 85\prime;\\
&\prime 45\prime;\prime\ f9\prime;\prime\ 02\prime;\prime\ 7f\prime;\prime\ 50\prime;\prime\ 3c\prime;\prime\ 9f\prime;\prime\ a8\prime;
\end{aligned}
$$

$$'51';' a3';' 40';' 8f';' 92';' 9d';' 38';' f5';$$
$$'bc';' b6';' da';' 21';' 10';' ff';' f3';' d2';$$
$$'cd';' 0c';' 13';' ec';' 5f';' 97';' 44';' 17';$$
$$'c4';' a7';' 7e';' 3d';' 64';' 5d';' 19';' 73';$$
$$'60';' 81';' 4f';' dc';' 22';' 2a';' 90';' 88';$$
$$'46';' ee';' b8';' 14';' de';' 5e';' 0b';' db';$$
$$'e0';' 32';' 3a';' 0a';' 49';' 06';' 24';' 5c';$$
$$'c2';' d3';' ac';' 62';' 91';' 95';' e4';' 79';$$
$$'e7';' c8';' 37';' 6d';' 8d';' d5';' 4e';' a9';$$
$$'6c';' 56';' f4';' ea';' 65';' 7a';' ae';' 08';$$
$$'ba';' 78';' 25';' 2e';' 1c';' a6';' b4';' c6';$$
$$'e8';' dd';' 74';' 1f';' 4b';' bd';' 8b';' 8a';$$
$$'70';' 3e';' b5';' 66';' 48';' 03';' f6';' 0e';$$
$$'61';' 35';' 57';' b9';' 86';' c1';' 1d';' 9e';$$
$$'e1';' f8';' 98';' 11';' 69';' d9';' 8e';' 94';$$
$$'9b';' 1e';' 87';' e9';' ce';' 55';' 28';' df';$$
$$'8c';' a1';' 89';' 0d';' bf';' e6';' 42';' 68';$$
$$'41';' 99';' 2d';' 0f';' b0';' 54';' bb';' 16']"$$

Results retrieved about AES S-box properties by SAMT are as follows:

1. Fixed Points of S-box = 0
2. Opposite Fixed Points of S-box = 0
3. S-Box is Bijective
4. Hamming weight of all Boolean functions

$$\begin{bmatrix} 128 & 128 & 128 & 128 & 128 & 128 & 128 & 128 \end{bmatrix}$$

5. S-box is Balanced

6. Non-linearity of all Boolean functions

$$\begin{bmatrix}112 & 112 & 112 & 112 & 112 & 112 & 112 & 112\end{bmatrix}$$

7. Average Non-linearity of all Boolean functions = 112

8. Almost Bent Non-linearity value = 120

9. Perfect Non-linearity of all Boolean functions

$$\begin{bmatrix}124 & 144 & 132 & 120 & 136 & 132 & 140 & 128\end{bmatrix}$$

10. Non-linearity of S-box = 112

11. Maximum Non-linearity value for given Galois field = 120

12. Differential Branch Number = 3

13. Dynamic Distance Table

$$\begin{bmatrix}
0 & 6 & 2 & 6 & 8 & 6 & 2 & 2 \\
4 & 0 & 6 & 2 & 0 & 8 & 2 & 4 \\
0 & 4 & 0 & 8 & 4 & 0 & 2 & 2 \\
6 & 0 & 4 & 0 & 6 & 4 & 4 & 4 \\
4 & 6 & 0 & 0 & 2 & 6 & 0 & 6 \\
4 & 4 & 6 & 4 & 4 & 2 & 2 & 6 \\
2 & 4 & 4 & 4 & 2 & 4 & 4 & 4 \\
2 & 2 & 4 & 2 & 4 & 2 & 8 & 2
\end{bmatrix}$$

14. Dynamic Distances of Boolean functions

$$\begin{bmatrix}6 & 6 & 6 & 8 & 8 & 8 & 8 & 6\end{bmatrix}$$

15. Desired Avalanche value for Boolean function = 128

16. Avalanche value of Boolean functions

$$\begin{bmatrix} 128 & 116 & 124 & 116 & 144 & 116 & 132 & 132 \\ 136 & 128 & 116 & 124 & 128 & 144 & 124 & 120 \\ 128 & 136 & 128 & 144 & 120 & 128 & 132 & 132 \\ 140 & 128 & 136 & 128 & 116 & 120 & 136 & 136 \\ 136 & 140 & 128 & 128 & 132 & 116 & 128 & 116 \\ 136 & 136 & 140 & 120 & 120 & 132 & 132 & 116 \\ 124 & 136 & 136 & 120 & 132 & 120 & 136 & 136 \\ 132 & 124 & 136 & 124 & 136 & 132 & 144 & 132 \end{bmatrix}$$

17. Boolean functions results which satisfy Avalanche Criterion = 43

18. Avalanche Criterion Percentage of S-box = 67.1875

19. Dependence Matrix of S-box

$$\begin{bmatrix} 0.5000 & 0.4531 & 0.4844 & 0.4531 & 0.5625 & 0.4531 & 0.5156 & 0.5156 \\ 0.5313 & 0.5000 & 0.4531 & 0.4844 & 0.5000 & 0.5625 & 0.4844 & 0.4688 \\ 0.5000 & 0.5313 & 0.5000 & 0.5625 & 0.4688 & 0.5000 & 0.5156 & 0.5156 \\ 0.5469 & 0.5000 & 0.5313 & 0.5000 & 0.4531 & 0.4688 & 0.5313 & 0.5313 \\ 0.5313 & 0.5469 & 0.5000 & 0.5000 & 0.5156 & 0.4531 & 0.5000 & 0.4531 \\ 0.5313 & 0.5313 & 0.5469 & 0.4688 & 0.4688 & 0.5156 & 0.5156 & 0.4531 \\ 0.4844 & 0.5313 & 0.5313 & 0.4688 & 0.5156 & 0.4688 & 0.5313 & 0.5313 \\ 0.5156 & 0.4844 & 0.5313 & 0.4844 & 0.5313 & 0.5156 & 0.5625 & 0.5156 \end{bmatrix}$$

20. Desired SAC of a Boolean function = 1024

21. SAC values of Boolean functions

$$\begin{bmatrix} 1008 & 1020 & 1048 & 1040 & 1024 & 1032 & 1040 & 1060 \end{bmatrix}$$

22. Sum of Squares Indicator of Boolean functions

$$\begin{bmatrix} 133120 & 133120 & 133120 & 133120 & 133120 & 133120 & 133120 & 133120 \end{bmatrix}$$

23. Absolute Indicator of Boolean functions

$$\begin{bmatrix} 32 & 32 & 32 & 32 & 32 & 32 & 32 & 32 \end{bmatrix}$$

24. BIC Non-linearity criterion of S-box

$$\begin{bmatrix} 0 & 112 & 112 & 112 & 112 & 112 & 112 & 112 \\ 112 & 0 & 112 & 112 & 112 & 112 & 112 & 112 \\ 112 & 112 & 0 & 112 & 112 & 112 & 112 & 112 \\ 112 & 112 & 112 & 0 & 112 & 112 & 112 & 112 \\ 112 & 112 & 112 & 112 & 0 & 112 & 112 & 112 \\ 112 & 112 & 112 & 112 & 112 & 0 & 112 & 112 \\ 112 & 112 & 112 & 112 & 112 & 112 & 0 & 112 \\ 112 & 112 & 112 & 112 & 112 & 112 & 112 & 0 \end{bmatrix}$$

3.2.2 Guo Chen S-box properties

"An extended method for obtaining S-boxes based on three-dimensional chaotic Baker maps" [11].

S-box presented in this paper for the MATLAB code in decimal format is given below.

('161 85 129 224 176 50 207 177 48 205 68 60 1 160 117 46 130 124 203 58 145 14
115 189 235 142 4 43 13 51 52 19 152 153 83 96 86 133 228 136 175 23 109 252
236 49 167 92 106 94 81 139 151 134 245 72 172 171 62 79 77 231 82 32 238 22 63
99 80 217 164 178 0 154 240 188 150 157 215 232 180 119 166 18 141 20 17 97
254 181 184 47 146 233 113 120 54 21 183 118 15 114 36 253 197 2 9 165 132 204
226 64 107 88 55 8 221 65 185 234 162 210 250 179 61 202 248 247 213 89 101
108 102 45 56 5 212 10 12 243 216 242 84 111 143 67 93 123 11 137 249 170 27
223 186 95 169 116 163 25 174 135 91 104 196 208 148 24 251 39 40 31 16 219
214 74 140 211 112 75 190 73 187 244 182 122 193 131 194 149 121 76 156 168
222 34 241 70 255 229 246 90 53 225 100 30 37 237 103 126 38 200 44 209 42 29

41 218 71 155 78 125 173 28 128 87 239 3 191 158 199 138 227 59 69 220 195 66
192 230 198 26 159 6 127 201 144 206 98 33 35 7 105 147 57 110′)

MATLAB tool compute following results:

1. Fixed Points of S-box = 0
2. Opposite Fixed Points of S-box = 1
3. S-box is Bijective
4. Hamming weight of all Boolean functions

$$\begin{bmatrix}128 & 128 & 128 & 128 & 128 & 128 & 128 & 128\end{bmatrix}$$

5. S-box is Balanced
6. Non-linearity of all Boolean functions

$$\begin{bmatrix}100 & 100 & 104 & 100 & 106 & 106 & 102 & 106\end{bmatrix}$$

7. Average Non-linearity of all Boolean functions = 103
8. Almost Bent Non-linearity value = 120
9. Perfect Non-linearity of all Boolean functions

$$\begin{bmatrix}144 & 128 & 144 & 124 & 140 & 140 & 140 & 128\end{bmatrix}$$

10. Non-linearity of S-box = 104
11. Maximum Non-linearity value for given Galois field = 120
12. Differential Branch Number = 3

13. Dynamic Distance Table

$$\begin{bmatrix} 0 & 2 & 0 & 14 & 8 & 2 & 0 & 8 \\ 2 & 10 & 4 & 6 & 10 & 4 & 6 & 6 \\ 0 & 2 & 0 & 4 & 4 & 2 & 2 & 4 \\ 10 & 4 & 4 & 2 & 6 & 0 & 4 & 4 \\ 0 & 6 & 8 & 6 & 6 & 4 & 2 & 4 \\ 2 & 6 & 0 & 2 & 2 & 4 & 8 & 2 \\ 8 & 2 & 4 & 8 & 2 & 2 & 2 & 2 \\ 4 & 6 & 6 & 0 & 10 & 4 & 6 & 6 \end{bmatrix}$$

14. Dynamic Distances of Boolean functions

$$\begin{bmatrix} 10 & 10 & 8 & 14 & 10 & 4 & 8 & 8 \end{bmatrix}$$

15. Desired Avalanche value for Boolean function = 128

16. Avalanche value of Boolean functions

$$\begin{bmatrix} 128 & 132 & 128 & 156 & 144 & 132 & 128 & 144 \\ 132 & 148 & 136 & 116 & 108 & 120 & 116 & 140 \\ 128 & 132 & 128 & 120 & 136 & 124 & 132 & 136 \\ 148 & 136 & 120 & 132 & 140 & 128 & 136 & 120 \\ 128 & 140 & 112 & 140 & 116 & 120 & 132 & 136 \\ 124 & 116 & 128 & 124 & 132 & 120 & 112 & 132 \\ 112 & 132 & 120 & 112 & 124 & 124 & 132 & 124 \\ 136 & 140 & 116 & 128 & 108 & 136 & 116 & 116 \end{bmatrix}$$

17. Boolean functions results which satisfy Avalanche Criterion = 38

18. Avalanche Criterion Percentage of S-box = 59.3750

19. Dependence Matrix of S-box

$$\begin{bmatrix} 0.5000 & 0.5156 & 0.5000 & 0.6094 & 0.5625 & 0.5156 & 0.5000 & 0.5325 \\ 0.5156 & 0.5781 & 0.5313 & 0.4531 & 0.4219 & 0.4688 & 0.4531 & 0.5469 \\ 0.5000 & 0.5156 & 0.5000 & 0.4688 & 0.5313 & 0.4844 & 0.5156 & 0.5313 \\ 0.5781 & 0.5313 & 0.4688 & 0.5156 & 0.5469 & 0.5000 & 0.5313 & 0.4688 \\ 0.5000 & 0.5469 & 0.4375 & 0.5469 & 0.4531 & 0.4688 & 0.5156 & 0.5313 \\ 0.4844 & 0.4531 & 0.5000 & 0.4844 & 0.5156 & 0.4688 & 0.4375 & 0.5156 \\ 0.4375 & 0.5156 & 0.4688 & 0.4375 & 0.4844 & 0.4844 & 0.5156 & 0.4844 \\ 0.5313 & 0.5469 & 0.4531 & 0.5000 & 0.4219 & 0.5313 & 0.4531 & 0.4531 \end{bmatrix}$$

20. Desired SAC of a Boolean function = 1024

21. SAC values of Boolean functions

$$\begin{bmatrix} 1092 & 1016 & 1036 & 1060 & 1024 & 988 & 980 & 996 \end{bmatrix}$$

22. Sum of Squares Indicator of Boolean functions

$$\begin{bmatrix} 199552 & 201856 & 181120 & 222976 & 196096 & 170368 & 202240 & 192640 \end{bmatrix}$$

23. Absolute Indicator of boolean functions

$$\begin{bmatrix} 64 & 64 & 64 & 72 & 64 & 72 & 72 & 80 \end{bmatrix}$$

24. BIC Non-linearity criterion of S-box

$$\begin{bmatrix} 0 & 102 & 104 & 102 & 106 & 104 & 100 & 102 \\ 102 & 0 & 104 & 100 & 106 & 102 & 106 & 100 \\ 104 & 104 & 0 & 104 & 108 & 108 & 102 & 100 \\ 102 & 100 & 104 & 0 & 108 & 100 & 100 & 108 \\ 106 & 106 & 108 & 108 & 0 & 106 & 98 & 100 \\ 104 & 102 & 108 & 100 & 106 & 0 & 104 & 102 \\ 100 & 106 & 102 & 100 & 98 & 104 & 0 & 102 \\ 102 & 100 & 100 & 108 & 100 & 102 & 102 & 0 \end{bmatrix}$$

3.2.3 Faith and Ahmet S-box properties

"A method for designing strong S-Boxes based on chaotic Lorenz system" [36].

S-box presented in this paper is given below in decimal format of MATLAB.

('60 215 166 47 119 212 85 136 117 65 238 242 182 39 229 143 31 129 128 218 8 27 99 45 241 179 187 73 237 138 15 203 227 205 216 144 202 49 130 254 131 81 148 178 127 121 221 133 13 230 92 48 188 199 58 116 44 43 137 153 34 112 231 103 67 204 211 206 152 118 96 57 77 126 74 50 244 253 164 226 6 10 36 38 94 98 59 184 115 170 232 7 162 68 150 248 72 167 159 177 105 142 56 93 114 192 249 90 84 102 154 222 134 125 141 183 185 169 87 189 156 155 217 197 201 89 2 9 228 186 240 173 195 104 100 101 29 33 252 236 18 193 26 213 250 55 176 95 20 146 17 1 219 139 79 132 194 61 14 207 53 62 180 255 63 174 69 35 32 42 28 4 124 19 75 23 147 80 54 200 158 165 120 140 190 11 220 157 210 106 145 107 239 40 246 91 243 5 151 111 214 37 25 233 82 86 21 245 76 172 22 78 122 198 30 224 168 209 225 110 64 181 251 208 88 71 235 109 51 108 161 0 191 223 247 171 149 196 66 113 123 41 3 70 163 175 135 16 12 234 97 83 160 24 52 46')

MATLAB code compute these results:

1. Fixed Points of S-box = 1

2. Opposite Fixed Points of S-box = 1

3. S-box is not Bijective

4. Hamming weight of all Boolean functions

$$\begin{bmatrix} 128 & 128 & 128 & 128 & 128 & 128 & 128 & 128 \end{bmatrix}$$

5. S-box is Balanced

6. Non-linearity of all Boolean functions

$$\begin{bmatrix} 104 & 104 & 102 & 104 & 102 & 106 & 100 & 104 \end{bmatrix}$$

7. Average Non-linearity of all Boolean functions = 103.2500

8. Almost Bent non-linearity = 120

9. Perfect Non-linearity of all Boolean functions

$$\begin{bmatrix} 128 & 148 & 132 & 120 & 112 & 124 & 120 & 132 \end{bmatrix}$$

10. Maximum Non-linearity value for given Galois field = 120

11. Non-linearity of S-box = 104

12. Differential Branch Number = 3

13. Dynamic Distance Table

$$\begin{bmatrix} 6 & 6 & 4 & 0 & 4 & 4 & 8 & 6 \\ 4 & 2 & 6 & 2 & 2 & 0 & 6 & 2 \\ 6 & 4 & 4 & 2 & 6 & 4 & 4 & 6 \\ 6 & 2 & 10 & 2 & 4 & 0 & 2 & 4 \\ 6 & 4 & 6 & 2 & 4 & 4 & 10 & 8 \\ 2 & 6 & 2 & 4 & 2 & 0 & 4 & 0 \\ 0 & 0 & 4 & 6 & 2 & 4 & 4 & 12 \\ 0 & 0 & 8 & 4 & 6 & 2 & 2 & 10 \end{bmatrix}$$

14. Dynamic Distances of Boolean functions

$$\begin{bmatrix} 6 & 6 & 10 & 6 & 6 & 4 & 10 & 12 \end{bmatrix}$$

15. Desired Avalanche value for Boolean function = 128

16. Avalanche value of Boolean functions

$$\begin{bmatrix} 116 & 140 & 120 & 128 & 136 & 120 & 144 & 116 \\ 120 & 124 & 116 & 124 & 124 & 128 & 140 & 124 \\ 116 & 136 & 136 & 124 & 140 & 136 & 136 & 116 \\ 140 & 132 & 148 & 132 & 120 & 128 & 124 & 136 \\ 140 & 120 & 140 & 132 & 120 & 136 & 148 & 112 \\ 132 & 140 & 132 & 136 & 132 & 128 & 120 & 128 \\ 128 & 128 & 136 & 116 & 132 & 120 & 136 & 152 \\ 128 & 128 & 144 & 136 & 116 & 124 & 124 & 108 \end{bmatrix}$$

17. Boolean functions results which satisfy Avalanche Criterion = 39

18. Avalanche Criterion Percentage of S-box = 60.9375

19. Dependence Matrix of S-box

$$\begin{bmatrix} 0.4531 & 0.5469 & 0.4688 & 0.5000 & 0.5313 & 0.4688 & 0.5625 & 0.4531 \\ 0.4688 & 0.4844 & 0.4531 & 0.4844 & 0.4844 & 0.5000 & 0.5469 & 0.4844 \\ 0.4531 & 0.5313 & 0.5313 & 0.4844 & 0.5469 & 0.5313 & 0.5313 & 0.4531 \\ 0.5469 & 0.5156 & 0.5781 & 0.5156 & 0.4688 & 0.5000 & 0.4844 & 0.5313 \\ 0.5469 & 0.4688 & 0.5469 & 0.5156 & 0.4688 & 0.5313 & 0.5781 & 0.4375 \\ 0.5156 & 0.5469 & 0.5156 & 0.5313 & 0.5156 & 0.5000 & 0.4688 & 0.5000 \\ 0.5000 & 0.5000 & 0.5313 & 0.4531 & 0.5156 & 0.4688 & 0.5313 & 0.5938 \\ 0.5000 & 0.5000 & 0.5625 & 0.5313 & 0.4531 & 0.4844 & 0.4844 & 0.4219 \end{bmatrix}$$

20. Desired SAC of a Boolean function = 1024

21. SAC values of Boolean functions

$$\begin{bmatrix} 1020 & 1000 & 1040 & 1060 & 1048 & 1048 & 1048 & 1008 \end{bmatrix}$$

22. Sum of Squares Indicator of Boolean functions

$$\begin{bmatrix} 182272 & 187264 & 182272 & 212992 & 190720 & 173056 & 208384 & 206464 \end{bmatrix}$$

23. Absolute Indicator of boolean functions

$$\begin{bmatrix} 64 & 72 & 80 & 64 & 56 & 56 & 80 & 64 \end{bmatrix}$$

24. BIC Non-linearity criterion of S-box

$$\begin{bmatrix} 0 & 106 & 104 & 102 & 104 & 106 & 108 & 102 \\ 106 & 0 & 104 & 100 & 104 & 102 & 108 & 104 \\ 104 & 104 & 0 & 106 & 104 & 102 & 104 & 102 \\ 102 & 100 & 106 & 0 & 100 & 100 & 104 & 102 \\ 104 & 104 & 104 & 100 & 0 & 102 & 106 & 104 \\ 106 & 102 & 102 & 100 & 102 & 0 & 104 & 104 \\ 108 & 108 & 104 & 104 & 106 & 104 & 0 & 106 \\ 102 & 104 & 102 & 102 & 104 & 104 & 106 & 0 \end{bmatrix}$$

3.3 Comparison of Results with SET-tool

The results calculated by SAMT were compared with the results obtained by SET-tool for the S-box of AES.

Properties of S-box	**SET-tool**	**SAMT**
Balanced	yes	yes
Bijective	yes	yes
Fixed points	0	0
Opposite fixed points	1	0
Non-linearity	112	112
Sum of square indicator	133120	133120
Absolute indicator	32	32
SAC	not satisfied	not satisfied
Non-linearity boolean functions	not available	calculated
Perfect Non-linearity boolean functions	not available	calculated
Differential Branch Number	not available	3
Dynamic distances boolean functions	not available	calculated
Avalanche effect boolean functions	not available	calculated
Avalanche effect percentage	not available	67%
Dependence Matrix of S-box	not available	calculated
BIC Non-linearity	not available	112

Table 3.1: AES S-box results with SET tool and SAMT

3.4 Conclusion

S-boxes and boolean functions play an important role in many algorithms of symmetric key cryptography. Often, we use some already known and widely used S-boxes or at least the logics behind them are reused (as in the case Rakaposhi Boolean function which use the same irreducible polynomial as AES) but there is always a requirement for the use of new S-boxes or Boolean functions. Hence we believe this tool can help cryptographic researchers to analyze the behavior of S-Boxes at a deeper level. With this MATLAB code, user will be able to execute desired properties not only for complete S-box but also for its all boolean functions. SAMT also provides a large space with respect to Galois field $GF(2^n)$ for $2 \leq n \leq 20$ so it will be a great tool in future for analysing bigger S-boxes than in present use $GF(2^8)$. MATLAB is a user friendly software and it provides a big space of built in functions and routines which can be used as a part of this code where required.

There are some aspects that we want to investigate in the further development of this tool. First one is to add more relevant cryptographic properties.

1. XOR Table
2. Algebraic Immunity
3. Correlation Immunity

These properties would make further enhancement in the analysis. We consider also an improvement in the execution speed of the code where we want to achieve the minimum time for the calculation of all properties of an S-box in $GF(2^n)$ for $n \geq 8$. This tool will be designed in such a way that it can analyze the properties of S-boxes with different input and output Galois fields.

Bibliography

[1] C. M. Adams, "On immunity against biham and shamir's differential cryptanalysis". *Information Processing Letters*, 41(2):77–80, 1992.

[2] M. Ahmad, D. Bhatia, & Y. Hassan, "A novel ant colony optimization based scheme for substitution box design". *Procedia Computer Science*, 57:572–580, 2015.

[3] J. A. Alvarez-Cubero & P. J. Zufiria, "A c++ class for analysing vector boolean functions from a cryptographic perspective". In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–9. IEEE, 2010.

[4] E. R. Berlekamp, "Factoring polynomials over finite fields". *Bell System Technical Journal*, 46(8):1853–1859, 1967.

[5] P. Bhadauriya, F. Suthar, & S. Chaudhary, "A novel technique for secure communication in cryptography".

[6] J. Blömer & J. P. Seifert, "Fault based cryptanalysis of the advanced encryption standard (aes)". In *International Conference on Financial Cryptography*, pages 162–181. Springer, 2003.

[7] E. Brow. "Algebraic geometry", 2010.

[8] C. Carlet, "Boolean functions for cryptography and error correcting codes". *Boolean Methods and Models*, 2006.

[9] C. Carlet, "Vectorial boolean functions for cryptography". *Boolean models and methods in mathematics, computer science, and engineering*, 134:398–469, 2010.

[10] C. Carlet & C. Ding, "Nonlinearities of s-boxes". *Finite fields and their applications*, 13(1):121–135, 2007.

[11] G. Chen, Y. Chen, & X. Liao, "An extended method for obtaining s-boxes based on three-dimensional chaotic baker maps". *Chaos, Solitons & Fractals*, 31(3):571–579, 2007.

[12] J. A. Clark, J. L. Jacob, S. Maitra, & P. Stănică, "Almost boolean functions: The design of boolean functions by spectral inversion". *Computational Intelligence*, 20(3):450–462, 2004.

[13] N. T. Courtois, "An improved differential attack on full gost". In *The New Codebreakers*, pages 282–303. Springer, 2016.

[14] J. Daemen & V. Rijmen, "The design of Rijndael: AES-the advanced encryption standard". Springer Science & Business Media, 2013.

[15] J. Daemen, R. Govaerts, & J. Vandewalle, "Correlation matrices". In *International Workshop on Fast Software Encryption*, pages 275–285. Springer, 1994.

[16] W. Diffie & M. E. Hellman, "Special feature exhaustive cryptanalysis of the nbs data encryption standard". *Computer*, 10(6):74–84, 1977.

[17] I. R. Dragomir & M. Lazăr, "Generating and testing the components of a block cipher". In *Electronics, Computers and Artificial Intelligence (ECAI), 2016 8th International Conference on*, pages 1–4. IEEE, 2016.

[18] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms". *IEEE transactions on information theory*, 31(4):469–472, 1985.

[19] H. Feistel, "Cryptography and computer privacy". *Scientific american*, 228 (5):15–23, 1973.

[20] R. Forrié, "The strict avalanche criterion: spectral properties of boolean functions and an extended definition". In *Conference on the Theory and Application of Cryptography*, pages 450–468. Springer, 1988.

[21] J. Gordon & H. Retkin, "Are big s-boxes best?". In *Workshop on Cryptography*, pages 257–262. Springer, 1982.

[22] H. Heys et al., "Selected Areas in Cryptography: 6th Annual International Workshop, SAC'99 Kingston, Ontario, Canada, August 9-10, 1999 Proceedings", volume 1758. Springer Science & Business Media, 2000.

[23] K. J. Horadam, "Hadamard matrices and their applications: Progress 2007–2010". *Cryptography and Communications*, 2(2):129–154, 2010.

[24] I. Hussain & T. Shah, "Literature survey on nonlinear components and chaotic nonlinear components of block ciphers". *Nonlinear Dynamics*, 74(4):869–904, 2013.

[25] P. D. Johnson Jr, G. A. Harris, & D. Hankerson, "Introduction to information theory and data compression". Chapman and Hall/CRC, 2003.

[26] J. B. Kam & G. I. Davida, "Structured design of substitution-permutation encryption networks". *IEEE Transactions on Computers*, (10):747–753, 1979.

[27] A. Kapoor. "Customizable public key infrastructure and development tool for same", Oct. 27 2009. US Patent 7,610,484.

[28] L. R. Knudsen & M. Robshaw, "The block cipher companion". Springer Science & Business Media, 2011.

[29] F. Lafitte, "The boolfun package: Cryptographic properties of boolean functions". 2012.

[30] S. A. Mahadevan, "Low power implementation of an AES 128-bit encryption". The University of Texas at San Antonio, 2015.

[31] P. P. Mar & K. M. Latt, "New analysis methods on strict avalanche criterion of s-boxes". *World Academy of Science, Engineering and Technology*, 48(150-154):25, 2008.

[32] W. Meier & O. Staffelbach, "Nonlinearity criteria for cryptographic functions". In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 549–562. Springer, 1989.

[33] S. Mister & C. Adams, "Practical s-box design". In *Workshop on Selected Areas in Cryptography, SAC*, volume 96, pages 61–76, 1996.

[34] K. Moeen, "Progressive product reduction for polynomial basis multiplication over GF (3m)". PhD thesis, 2016.

[35] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, & E. Roback, "Report on the development of the advanced encryption standard (aes)". *Journal of Research of the National Institute of Standards and Technology*, 106(3):511, 2001.

[36] F. Özkaynak & A. B. Özer, "A method for designing strong s-boxes based on chaotic lorenz system". *Physics Letters A*, 374(36):3733–3738, 2010.

[37] S. Picek, L. Batina, D. Jakobović, B. Ege, & M. Golub, "S-box, set, match: a toolbox for s-box analysis". In *IFIP International Workshop on Information Security Theory and Practice*, pages 140–149. Springer, 2014.

[38] S. Picek, D. Jakobovic, J. F. Miller, L. Batina, & M. Cupic, "Cryptographic boolean functions: One output, many design criteria". *Applied Soft Computing*, 40:635–653, 2016.

[39] J. Pieprzyk & G. Finkelstein, "Permutations that maximize non-linearity and their cryptographic significance". North-Holland, Amsterdam, 1989.

[40] B. Preneel & A. BRAEKEN, "Cryptographic properties of boolean functions and s-boxes". Technical report, Technical report, Katholieke Universiteit Leuven, 2006.

[41] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, & J. Vandewalle, "Propagation characteristics of boolean functions". In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 161–173. Springer, 1990.

[42] V. Rijmen & J. Daemen, "Advanced encryption standard". *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pages 19–22, 2001.

[43] R. L. Rivest, A. Shamir, & L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems". *Communications of the ACM*, 21 (2):120–126, 1978.

[44] S. Sarkar & H. Syed, "Bounds on differential and linear branch number of permutations". In *Australasian Conference on Information Security and Privacy*, pages 207–224. Springer, 2018.

[45] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, T. Kohno, & M. Stay, "The twofish teams final comments on aes selection". *AES round*, 2, 2000.

[46] J. Seberry, X.-M. Zhang, & Y. Zheng, "GAC: The Criterion for Global Avalanche Characteristics of Cryptographic Functions". University of Wollongong. Department of Computing Science, 1994.

[47] N. Sklavos & X. Zhang, "Wireless security and cryptography: specifications and implementations". CRC press, 2007.

[48] M. Stanek, "On cryptographic properties of random boolean functions". *Journal of Universal Computer Science*, 4(8):705–717, 1998.

[49] W. Stein, "Sage mathematics software". *http://www. sagemath. org/*, 2007.

[50] G. Tang, X. Liao, & Y. Chen, "A novel method for designing s-boxes based on chaotic maps". *Chaos, Solitons & Fractals*, 23(2):413–419, 2005.

[51] Y. Tarannikov, P. Korolev, & A. Botev, "Autocorrelation coefficients and correlation immunity of boolean functions". In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 460–479. Springer, 2001.

[52] I. VERGİLİ & M. D. Yücel, "Avalanche and bit independence properties for the ensembles of randomly chosen n\times n s-boxes". *Turkish Journal of Electrical Engineering & Computer Sciences*, 9(2):137–146, 2001.

[53] Y. Wang, K.-W. Wong, C. Li, & Y. Li, "A novel method to design s-box based on chaotic map and genetic algorithm". *Physics Letters A*, 376(6-7): 827–833, 2012.

[54] A. Webster & S. E. Tavares, "On the design of s-boxes". In *Conference on the theory and application of cryptographic techniques*, pages 523–534. Springer, 1985.

[55] Y. Wei & E. Pasalic, "On the approximation of s-boxes via maiorana–mcfarland functions". *IET Information Security*, 7(2):134–143, 2013.

[56] G. Xu, G. Zhao, & L. Min, "A method for designing dynamical s-boxes based on discrete chaos map system". In *Communications, Circuits and Systems, 2009. ICCCAS 2009. International Conference on*, pages 876–880. IEEE, 2009.

[57] A. Youssef, S. Tavares, S. Mister, & C. Adams, "Linear approximation of injective s-boxes". *Electronics letters*, 31(25):2165–2165, 1995.

[58] Y. Zheng, "Systematic generation of cryptographically robust s-boxes". 1993.