ALGEBRRAIC CRYPTANALYSIS OF FEISTEL STRUCTURE BASED LIGHT WEIGHT BLOCK CIPHERS

By

Lt Cdr Nasir Mehmood PN

A thesis submitted to the faculty of Information Security Department Military College of Signals, National University of Sciences and Technology, Pakistan, in partial fulfillment of the requirements for the degree of MS in Information Security

June 2014

SUPERVISOR CERTIFICATE

It is to certify that Final Copy of Thesis has been evaluated by me, found as per format and error free.

Dated_____ **Dr. Mehreen Afzal**

Abstract

Light weight block cipher is a new trend in cipher design that is aimed at providing a trade-off between security and efficiency for resource constrained special purpose applications like RFID-tags, sensor nodes and smart card. Consequently, various cryptanalytic techniques are also taken into account to gauge very carefully the security of these light-weight ciphers. Algebraic cryptanalysis has been extensively applied to break many real world stream ciphers; however, exploitation of its potential against block ciphers is a grey area of research. In algebraic methods of block ciphers cryptanalysis, linear and nonlinear components; separately cipher and key scheduling, are modeled into systems of algebraic equations. These are then combined to determine the complex system of equations that completely describe the entire cipher. Solution of such systems, where possible, gives the key or plaintext.

In this thesis, basic concept behind algebraic technique, light weight block ciphers and their algebraic cryptanalysis has been discussed. Due presence of nonlinear component i.e. S-box, in cipher design, resistivity of block ciphers against algebraic attacks lies in the S-box. This research also describes a step by step methodology to model any S-box in to system of linearly independent Multivariate Quadratic (MQ) equations. Initially, Proof of Concept (PoC) on a simpler 3x3 S-box has been implemented. Then targeted feistel structure based light weight block ciphers have been analyzed with respect to their resistivity against algebraic attacks. A simple algebraic representation of 32 round LBlock in terms of 2628 variables, 8928 equations and 43,908 monomials, 33 rounds of $SEA_{48,8}$ in terms of 3216 variables, 10,560 equations in 34,320 monomials and $SEA_{96,8}$ in terms of 6432 variables, 21,120 equations and 68, 568 monomials have been given. Moreover, it has also been shown that XSL attack doesn't pose any threat to either LBlock or SEA. In addition, feasibility about applicability of cube attack in combination with algebraic attack has also been undertaken.

A software tools has also been developed using Maple/C-sharp that can give algebraic representation of any lower order S-box. Developed tool can be utilized in S-box design as well as in algebraic cryptanalysis of other block ciphers.

DEDICATION

To My Loving Father

May Allah reside his soul at the most peaceful place in heaven. Amen

ACKNOWLEDGEMENT

All Glory to Almighty Allah, the most gracious and the most merciful, Master of the Day of Judgment, without His blessings and kindness, it's impossible to achieve anything.

First of all I would like to thank my honorable supervisor Dr. Mehreen Afzal, for her remarkable patience, gracious, generous, overwhelming and untiring support and guidance throughout the work. Without her support and guidance this thesis could not be completed in time. Besides being an outstanding supervisor/teacher, she is also an amazing human being blessed with nice temperament commensurate to the teacher's role.

I am also thankful to my guidance committee members Lt. Col. Dr. Adil Masood Saddiqui, EE Dept, Cdr. Dr. Muhammad Farhan PN and Lect. Main Waseem Iqbal, IS Deptt for their able guidance and support.

I am extremely thankful to my Head of Department Lt.Col.Dr. Baber Aslam and his team for administrative help and support.

I would also like to thank anonymous people for rendering guidance via emails, my family for persistent care and support, my friend Lt Cdr Faisal Ahmed Khan PN, seniors specially Capt Syed Muhammad Babur TI (M) PN and Lt Cdr(R) Muhammad Azam PN for consistent moral support and encouragement in difficult times to complete this work otherwise this would not have been possible to complete it due hectic service routine.

TABLE OF CONTENTS

1 Introduction……..…..1

1.1 Overview.....1

1.2 Need for Research.....3

1.3 Problem Statement.....3

1.4 Objectives 4

1.5 Research Methedolgy.....4

1.6 Thesis Organisation 5

2 **Light Weight Block Ciphers and Algebraic Cryptanalysis – Literature Review.....6**

2.1 Introduction.....6

2.2 Block Ciphers 6

2.2.1 Light Weight Block Ciphers 7

2.2.2 Design Rationale of Light Weight Block Ciphers…….....8

2.3 Algebraic Cryptanalysis and Block Cipher.....8

2.3.1 Concept and Methodology of Algebraic Cryptanalysis Technique in Block Cipher…….....…9

2.3.1.1 Writing Systems of Equations 10

2.3.1.2 Solution of Systems of Equations 11

2.4 Techniques used for Solution of Multivariate Equations..... 12

2.4.1 Grobner.Bases.....12

2.4.2 Linearization.....13

2.4.3 Relinearisation.....14

2.4.4 XL Algorithem.....14

2.4.5 ElimLin…….....14

2.4.6 SAT Solvers.....15

2.4.6.1 SAT Method.....16

2.4.6.2 UNSAT Method.....16

2.5 Conclusion.....17

3 Algebraic Representation and Analysis of Light Weight Block Ciphers.....18

3.1 Introduction.....18

3.2 Modeling Light Weight Block Ciphers into Systems of Equations.....19

3.2.1 Step by Step Methodology for describing any S-box into Systems of Equations.....19

3.2.1.1 Define Monomials Based on Equations Degree Required to be Obtained - Step1.....21

3.2.1.2 Matrix Generation – Step 2.....21

3.2.1.3 Gaussian Elimination of the Generated Matrix – Step 3.....22

3.2.1.4 XOR Mod 2 Monomials and Output Results – Step 4.....23

3.2.1.5 Generate Matrix from the Equations – Step 5.....23

3.2.1.6 Gaussian Elimination Mode 2 of *l x M* Matrix – Step 6.....24

3.2.1.7 Verification of Equations for all Input/output of S-box 24

3.2.1.8 Some Findings Based on Analysis of Methodology for Formulating S-Box Equations 25

3.2.1.9 Comparison of Results Obtained with Published Results.....27

3.2.2 Equations Systems for Linear Components.....27

3.2.3 Obtaining Equations Systems for the Complete Cipher Scheme.....28

3.3 Complexity of XSL Attack.....28

3.4 Conclusion.……..30

4 Algebraic Representation and Analysis of LBlock Cipher.....31

4.1 Introduction……31

4.2 LBlock Light Weight Block Cipher……...32

4.2.1 Structure of LBlock Encryption / Decryption Algorithm…….....32

4.2.2 Key Scheduling……34

4.3 Results of Algebraic Equations for S Boxes……35

4.3.1 Some Findings on LBlock S-Box Equation…….……38

4.4 Constructing Equations Systems for Complete LBlock over GF(2).....38

4.4.1 Equations Systems over LBlock Encryption/Decryption Algorithm….....39

4.5 Complexity of XSL Attack over LBlock……..41

4.6 Conclusion……41

5 Algebraic Representation and Analysis of Scalable Encryption Algorithm…….....42

5.1 Introduction……..42

5.2 Scalable Encryption Algorithm for Resource Constrained Applications…….42

5.2.1 Structure of SEA Encryption/Decryption Algorithm…….…..44

5.2.2 Structure of SEA Key Scheduling Algorithm…….….45

5.3 Results of Algebraic Equations for S Box…...…...……...46

5.3.1 Some Findings over SEA S-box Equations..…….47

5.4 Equations Systems for Complete Cipher over GF (2)...…….48

5.5 Complexity of XSL Attack over SEA……..……..49

5.6 Conclusion ……...……...50

6 AIDA/Cube Attack and Algebraic Attack….....51

6.1 Introduction…….51

6.2 Cube Attack……. …….....51

6.2.1 Definition 7.1.……..…….....51

6.2.2 Definition 7.2 ……...52

6.2.3 The Pre-Processing Phase……..52

6.2.4 Online Phase……..54

6.3 Combining Cube and Algebraic Attacks on Light Weight Block Ciphers……..54

6.4 Conclusion……55

7 Conclusion and Future Work.....56

7.1 Introduction……56

7.2 Overview of the Research …….....56

7.3 Achievements…….....57

7.4 Future Work…..……...57

7.5 Conclusion…….....58

Appendix-A Output Combination for 3x3 Sbox with XOR Sum Zero…….. 59

Appendix-B LBlock S-Box Equations.....…….....63

Appendix-C Statistical Analysis of Individual S-box equations –LBlock ……...72

Appendix-D User Manual for Software Tool…… .75

BIBLIOGRAPHY 76

LIST OF FIGURES

Figure Number **Page**

Fig 4.1 Figure 5.1 Luban Lock……31

Fig 4.2 Encryption/Decryption Algorithm of LBlock…….33

Fig 4.3 F-Function of LBlock……..33

Fig 4.4 Key Scheduling of LBlock……...35

Fig 5.1 Encryption/Decryption Algorithm of SEA……..44

Fig 5.2 Key Scheduling Algorithm of SEA.……45

LIST OF TABLES

Table Number **Page**

Table 3.1 Contents of a 3 x 3 S-box……...20

Table 3.2 Matrix Generation for 3 x 3 S-box……22

Table 3.3 3 x 3 S-box Matrix in Reduced Row Echelon Form…… 23

Table 3.4 Statistical Analysis of Monomials in S-Box Equations……...26

Table 3.5 Comparison of Results for S-box Equation – Monomials……..27

Table 4.1 General specifications of LBlock…….....32

Table 4.2 Contents of LBlock S-boxes…….36

Table 4.3 Summary of LBlock Sbox Equations - Degree of Monomials…….36

Table 4.4 Statistical Analysis of LBlock S-box Equations – I/O Degree of Variables……..37

Table 4.5 Systems of Equations for LBlock over GF (2)……..…….40

Table 5.1 General Specifications of SEA…….……43

Table 5.2 Contents of S-box in SEA…….45

Table 5.3 Statistical Analysis of SEA S-box Equations...……47

Table 5.4 Systems of Equations for SEA over GF (2)……..…....49

Table 6.1 Master Polynomial *p* Expressed as Cube Equation……...53

LIST OF ABBREVIATIONS

ANF	Algebraic Normal Form
AES	Advance Encryption Standard
CP	Chosen Plaintext
CNF	Conjunctive Normal Form
CAS	Chinese Academy of Science
DES	Data Encryption Standard
ELIMLIN	Eliminate Linear
FPGA	Field Programmable Gate Array
GF	Galois Field
GE	Gate Equivalent
I/O	Input/Output
KP	Known Plaintext
LBlock	Luban Lock or Lightweight Block
LSW	Least Significant Word
MQ	Multivariate Quadratic Equations
NP	Non-deterministic Polynomial time
RFID	Radio Frequency Identification
SAT	Boolean Satisfiability Problem
SPN	Substitution Permutation Network
S-Box	Substitution Box
SEA	Scalable Encryption Algorithm
WRT	With Respect To

WSN	Wireless Sensor Node
XL	Extended Linearization
XSL	eXtended Sparse Linearization

Chapter 1

Introduction

1.1 Overview

In current era of "Information Age", information exploitation for own's advantages and its denial to the adversary is mission critical. Thus on one hand, securing the information and communication system with robust crypto systems is the prime objective. While on the other hand, main focus remains on testing crypto systems security with an aim to find vulnerabilities and/or improve the system design.

Ciphers are the class of crypto systems that provides data confidentiality in transit. In the last decade, a recent concept of light weight block ciphers have emerged for provisioning requisite security coupled with desired efficiency for extremely resource constrained applications like Radio Frequency Identification (RFID) tag, Wireless Sensors Node (WSN) and smart card etc. Block ciphers are also broadly categorized into feistel structure based and Substitution Permutation Network (SPN). Light weight block ciphers based on feistel structures are current generation ciphers that are aimed at providing higher level of security in comparison to stream ciphers for resource constraint applications, while consuming minimal resources. As a rule of thumb, all newly proposed ciphers are also critically analyzed against at least known cryptanalytic attacks for ascertaining the ciphers security and analyzing its structure for its effective and secure utilization in real time application.

Algebraic cryptanalysis is a technique that works by converting cipher system in to polynomial system of equations, then solving this system of equations to extract either key or plain text [1, 2, 3, 4, 5]. It takes advantage of

multivariate systems of equations in Algebraic Normal Form (ANF) that represent the targeted ciphers [6]. In algebraic methods of block ciphers cryptanalysis, linear and nonlinear components; separately encryption/decryption algorithm and key scheduling, are individually modeled into systems of equations. These equations are then combined to make a larger system of equations that completely describe the pertinent cipher. Since, block ciphers have nonlinear components; S-boxes, in their inherent design. Therefore, security of block ciphers against algebraic attacks lies in nonlinear components. Accordingly, modeling nonlinear components in terms of systems of equations dictates the cipher resistivity against algebraic attacks.

After the algebraic representation of block cipher has been obtained, the attacker then aims to solve the system of equations to determine unknown key (generally in a known plaintext settings, but quite often chosen plaintext is also used to reduce the time complexity of attack). The problem of solving large system of multivariate equations over a finite field is considered to be NP-Hard. Accordingly, this computational hardness peculiarity of solving large systems of multivariate equations forms the basis for security of most of the modern cryptographic systems against algebraic attacks. However, numerous algebraic techniques for solving complex system of multivariate equations have been proposed like finding Groebner Bases, Linearization, XL, ElimLin algorithm, Mix Integer Programming, SAT solvers etc. to break mostly stream ciphers based cryptosystems efficiently [4, 7, 8, 9, 10]. Among various techniques available for efficient solution of multivariate equations, SAT solvers have emerged as one of the recent yet very powerful tool. Moreover, if the system of equations is solved then attacker finds the unknown key. However, if the equations are very complex

and can't be solved still they depict the estimated cipher resistivity against algebraic attacks, as in case of AES [11, 12, 13].

1.2 Need for Research

Due recent boost in use of ubiquitous devices like RFID tags, smart cards and WSN nodes, there is an ever growing need to undertake threadbare security analysis of basic security primitives employed with respect to each and every of cryptanalysis. Algebraic cryptanalysis has enjoyed tremendous success against stream ciphers; however, exploitation of its real potential against block ciphers is a grey area for research. Although researchers have started analyzing block ciphers resistivity against algebraic attacks specifically after the emergence of algebraic representation of US NIST Advance Encryption Standard [13], however, lot of light weight block ciphers are yet to be algebraically analyzed. Few among these are LBlock [14], SEA [15], HIGHT [16], mCrypton [18], CGEN [19], PRESENT [20], DES Variants [21], LED [22], EPCBC [23], PICCOLO [24], PUFFIN [25] etc.

1.3 Problem Statement

There is a need to analyze cipher resistivity with respect to each and every cryptanalytic angel during the cipher's life cycle, so that vulnerabilities in its design can be exposed and its robustness can be gauged. Presently, no open source tool is available to algebraically analyze the block ciphers. Since nonlinear components in block cipher determine the ciphers resistivity against algebraic attacks. So there is a need to develop a software tool for algebraic representation and analysis of lower order S-boxes that are used in light weight ciphers. Light weight block cipher LBlock [14], design and developed by Chinese Academies of

Science (CAS), has been extensively analyzed against Cube, Linear, Differential, Boomerang, Related Key and Biclique cryptanalysis [26, 27, 28, 29, 30]. However, its algebraic representation and analysis is yet to be reported. Scalable Encryption Algorithm (SEA), an ultra light weight cipher designed by Belgium National Research (BNR), for extremely resource constrained applications [31, 32, 33, 34] has also been analyzed with respect to linear, differential, truncated differential cryptanalysis and square attacks [15], however, its algebraic representation and analysis is yet to be reported.

1.4 Objectives

Objective of this research is to undertake algebraic representation and analysis of feistel structure based light eight block ciphers; LBlock and SEA. This analysis has been undertaken after through literature review about applicability of algebraic cryptanalytic technique against block ciphers. A step by step methodology for algebraic representation of any S-box has been undertaken. Another objective is to develop a software tool that can give algebraic representation of any lower order S-boxes. Another academic objective assigned is to study about applicability of cube attack in comparison with algebraic attack on light weight structures.

1.5 Research Methodology

In order to undertake this research, the entire work has been divided into three main phases. In the first phase, literature review has been undertaken to build the concepts about algebraic cryptanalysis with reference to light weight block ciphers. Various tools and techniques applicable to algebraic cryptanalysis of block cipher has been described. In the second phase, a detailed procedure to transform any block cipher into systems of multivariate equations using a

systematic approach has been delineated. A step by step methodology for transforming lower order S-boxes into system of equations has been proposed. Implementation of proposed methodology on a prototype 3x3 S-box has been undertaken and it has been demonstrated that it gives better results than previously published. A generic software tool using Maple and C-sharp has also been developed that can analyze any lower order S-box. In the last phase, algebraic representation and analysis of LBlock and two version of SEA has been presented. Resistivity of both the ciphers against XSL attack has also been evaluated. In the end, feasibility study about applicability of Cube attack in combination with algebraic attack has also been undertaken.

1.6 Thesis Organization

This thesis report is organized into 7 chapters. Chapter 2 presents comprehensive literature review about the light weight block ciphers and algebraic cryptanalysis. It also discusses an overview of various computer algebra tools and techniques used for solution of complex system of equations. Chapter 3 describes the research methodology adopted in this thesis. It also contains a step by step procedure for formulating system of equations that completely describes any S-box with the help of a toy example. Then the methodology for algebraic representation of any block cipher has been discussed. Chapter 4 and Chapter 5 present results of algebraic representation against LBlock and SEA over GF(2) respectively. Resistivity of both the ciphers against XSL attack has also been determined. Chapter 6 describes the feasibility study about applicability of Cube attack in comparison with classical algebraic attack on block ciphers. Chapter 7 concludes the thesis.

Chapter 2

Light Weight Block Ciphers and Algebraic Cryptanalysis – Literature Review

2.1 Introduction

Light weight block ciphers are current generation child of traditional block ciphers. In this chapter, basics of block ciphers, light weight block ciphers and their design rationale has been discussed. Algebraic cryptanalytic technique in case of light weight block ciphers has also been discussed in detail.

The chapter has been divided into four main sections. Section 2.2 contains an overview of basic block ciphers, design principles and requirements leading to the advent of light weight block ciphers. Section 2.3 presents the core concept and methodology of algebraic cryptanalysis in case of block ciphers. Section 2.4 discusses most well known tools that are relevant for the solution of multivariate system equations as for as block ciphers cryptanalysis is concerned. Section 2.5 concludes the chapter.

2.2 Block Ciphers

Block ciphers are the class of cryptosystems that divides the plaintexts into chunk of blocks, and then use the encryption algorithm to generate cipher text. The core components of the block ciphers are substitution and permutations. Substitution generates confusion while permutation generates diffusions in the cipher. The substitution is the only nonlinear component and takes m bits as an input and gives n bits as an output. While permutation is the linear layer that usually permute the bits to diffuse the relationship of key into cipher text.

Moreover, block ciphers consist of may iterative rounds as per the specific cipher design criteria.

2.2.1 Light Weight Block Ciphers

Since the beginning of 21st century, use of resource constraints devices like RFID tag, smart card, wireless sensor nodes etc has gained a enormous boost mostly in security related special purpose applications. Almost all of us used such devices in our daily use. Since these application requires very efficient data exchange while consuming less resources but with an impregnable security. Therefore, a new trend of light weight cryptography has emerged in the recent past to meet the requirement of these special purpose applications in terms of security and efficiency. Nonetheless, stream ciphers had been employed in applications where resources and speed were the main concerns. However, since no comparison can be drawn between the level of security of a stream cipher and that of a block cipher, and traditional block ciphers like AES, Serpent etc can't be deployed in these applications so pressing requirement was felt to design light weight block ciphers. Light weight block ciphers have very efficient hardware implementation; consume very less resources while providing robust data confidentiality.

Light weight block ciphers are designed based on feistel as well as SPN structure. Both have their own pros and cons. However, feistel structure based ciphers schemes have always remained the prime attraction of cryptographic community due to certain advantage associated with them. These include smaller round function then SPN based ciphers, since only half of the block is processed in each round. Moreover, feistel schemes have inherent design feature to support a decryption function without any significant implementation cost. Accordingly,

numerous light weight block ciphers have been proposed that are extremely efficient and suit these special purpose applications. These includes feistel based ciphers like LBlock [14], SEA [15], DES Variants [21], GOST [35, 36], KASUMI [37], MTSUI [37], GFN based HIGHT [16], and SPN based ciphers PRINT [17], mCrypton [18], CGEN [19], PRESENT [20], and KATAN/KTANTAN [38].

2.2.2 Design Rationale of Light Weight Block Ciphers

Light weight block ciphers have few distinct peculiarities that not only separate them from traditional block ciphers but also drive the basic cipher design. Firstly, resource constrained applications operates on very small amount data, therefore, even relatively low throughput of light weight cipher is sufficient for targeted applications. Secondly, light weight ciphers are usually implemented in hardware environment for efficiency except in some cases a part of cipher is also implemented on 8 bit microcontroller. Thus efficient hardware implementation is another design rationale. Thirdly, light weight block cipher provides a tradeoff between security and performance, and it is assumed that moderate amount of security is sufficient. Lastly, it is well understood to the designer that the targeted application has very week computational ability, less memory and scanty power availability.

2.3 Algebraic Cryptanalysis and Block Cipher

The foundation stone for algebraic cryptanalysis was laid by C.E.Shanon, who related the security of cryptographic systems to the difficulty/complexity of solving set of algebraic equations that completely describe the targeted cipher [3]. Thus, in algebraic cryptanalysis, the cipher is modeled into set of multivariate

equations in ANF representation over a finite field usually GF (2) [39]. Generally, some crypto systems can be equivalently represented by multiple set of algebraic systems. The resultant systems of equations are quite often sparse, since practical implementation requires low Gate Equivalents (GE). Therefore, overall security of any crypto system can be gauged in terms of complexity in solving large system of multivariate equations. This problem of solving systems of multivariate equations is NP hard. However, several techniques for solution of complex system of equations have also been proposed [2, 3, 7, 40, 41, 42, 43]. Since its inception, algebraic cryptanalysis has been successfully applied against numerous stream ciphers. Despite enjoying tremendous success against stream ciphers based crypto systems, real potential of algebraic technique is yet to be exploited against block ciphers. However, the importance of this technique needs no emphasis since it is the only most effective attack against any crypto systems in real time scenario where only one or limited Known Plaintext (KP) is available with the attacker. Therefore, despite having nonlinear component in block cipher that swells the systems of equation as the number of rounds increases, recently, security evaluations of some block ciphers against algebraic attacks [4, 5, 35, 36, 37] has been published.

2.3.1 Concept and Methodology of Algebraic Cryptanalysis Technique in Block Cipher

It is important to note that actual potential of algebraic technique against block ciphers cryptanalysis is not fully explored mainly due difficulty in dealing with systems of equations that swells manifold with each round of cipher. And this is due to the presences of the S-boxes; the only source of nonlinearity in many block ciphers, and equations describing them are the main hurdle in solving

[37]. However, if the equations system have some regular structure besides being sparse, its manipulation becomes relatively easier, as in case of Rijndael and Serpent [44].

As a leading step not only to explore the uncharted waters but to advance the field of algebraic cryptanalysis against block ciphers, a UK based researcher published a cipher, namely Courtois Toy Cipher that was exclusively tailored to suit algebraic attacks [45, 46]. CTC had a simplified algebraic structure, random permutation of S-boxes, same key and block size and diffusion layer to achieve avalanche effect. Nicolas Courtois demonstrated to break the 255 bit block size, for 6 rounds in one hour on his note book PC through algebraic cryptanalysis [45, 46]. However, detailed methodology about the attack was not published with fear that it might endanger the security of AES, since it has typical algebraic structure. ‘‘In order to protect the United States government, the financial institutions, mobile phone operators, and hundreds of millions of other people that use AES, from criminals and terrorists, the exact description of the attack will for some time not be published. Public demonstrations of the effectiveness of the attack will be organized instead” [45]. Courtois also termed his attack as fast algebraic attack since it was based on adding additional intermediate state variables so that degree of equations should not increase with increase in round.

The core concept of algebraic cryptanalysis against block cipher revolves around two steps. These include, writing system of multivariate equations that describe the cipher, and finally solving these systems of equations to determine the key. Brief overview of these steps is described in ensuing paragraphs.

2.3.1.1 Writing Systems of Equations

The most tricky part is smartly modeling the cipher systems into systems of as simple equations as possible. The attacker starts with the nonlinear component i.e. S boxes, since these are the most pivotal as for as resistivity of block ciphers against algebraic attack is concerned. Modeling S-box completely in terms of systems of equations through exhaustive search is not feasible even for small S-box. Aim is to find linearly independent quadratic multivariate equations encompassing input/output variables of S-box that completely describes the pertinent S-box. Where possible, it is desirable to have sparse and over defined systems of equations due their ease in solving at later stages of attacks.

Next step is to target linear layers of the cipher, since these consists of simple linear operations like XOR, permutation, bit shifting etc. Therefore, writing systems of equations for linear layer is quite simple and straight forward. Finally, equations obtained from linear and nonlinear components are combined to form a system of equations that completely define the cipher. Moreover, in order to exploit systems of equation through algebraic attacks, it is important to make sure that degrees of equations don't increase drastically, as the cipher rounds goes up. Accordingly, algebraic complexity reduction; a technique based on the cipher design in general and S-box design in particular such that by guessing certain number of bits and determining some other bits, results in overall complexity reduction of equations system was proposed in GOST cryptanalysis [35]. Another technique is based on use of intermediate state variables for obtaining sparse systems of equations [45] that can be employed so that degree of equations don't swell with each round.

2.3.1.2 Solution of Multivariate Equations

After the block cipher's representation in terms of system of equations has been obtained, the next step is to solve them. The problem of solving MQ system of equations is NP hard. However, its complexity drops substantially when systems becomes overdefined i.e. having more number of equations than variables. This is an active area of research, and several tools/techniques for solution of systems of equations has been proposed [2, 3, 7, 40, 41, 42, 43], that shall be discussed in Section 2.4. Moreover, SAT solvers have emerged as a most powerful tool in solving these equations efficiently, while converting ANF equations into CNF form. It is pertinent to mention that complex system of equations is not always solvable, nonetheless, it still can describe the resistivity of cipher against algebraic attack as in the case of many ciphers like AES [11, 13, 44].

2.4 Techniques used for Solution of Multivariate Equations

Since the advent of algebraic method in cryptanalysis, various techniques has either exclusively been developed by the cryptographic community or few developed for other applications has been employed through improvisation. These includes Linearization and XL [7], XSL [44], DR [47], Zhuang-Zi [48], F4 [42], F5 [43], ElimLin [41], SAT solvers [40]. Some of the well reputed techniques or algorithm has been described in ensuing paragraphs:

2.4.1 Groebner Bases

This method was originally proposed by Bruno Buchburger in the pursuance of his doctorial research thesis in 1965 and he named it upon his research supervisor. Besides its utility in solving algebraic equations, this method has lot of other applications including in coding theory and robotics optimisation

etc. Basically groebner basis *G* can be defined as a specific type of generating subset of an ideal *I* in a certain polynomial ring *R*. The underlying approach in this method as for as solving systems of algebraic equations is that if we have equations $f1 = 0, \ldots, fn = 0$ so we calculate its reduced Groebner basis $G = \{g1, \ldots, gl\}$ for the polynomial ideal *I* generated by *hf1, . . . , fni*. Thus solution of polynomial equations $g1 = 0, \ldots, gl = 0$ can be equated to solving $f1 = 0, \ldots, fn = 0$. It is preferred to solve $g1 = 0, \ldots, gl = 0$ than $f1 = 0, \ldots, fn = 0$ due relative ease of solving.

Groebner basis is a broader approach, and on its underlying principle, various algorithm has been proposed like Buchberger F4 [42] and F5 [43] algorithms. However, the main drawback of these types of methods viz-a-viz block cipher cryptanalysis is during computational phase, the algorithms quite often crashes due low memory. This is only true for larger systems, but these methods are pretty faster than other methods when employed in case of smaller system of equations [49]. Groebner basis based F4 algorithm has been deployed in reduced round; up to 5, algebraic cryptanalysis of light weight block cipher PRINT [26].

2.4.2 Linearization

Linearization is a simpler technique but yet very effective in case of solving large and complex systems of polynomial equations. The idea is that for any given system of polynomials, each monomial can be replaced with a new variable. For example, if we have two degree monomial, so it can be replaced with a single degree variable. Resultantly, linearised system of polynomial equations is obtained. Solution of these linearised systems is checked against non-linearised original systems of equations. Overall efficacy of linearization

technique depends upon total number of linearly independent polynomials in the system. Ultimately, linearised system is then solved using any of the conventional algebra tools like Gaussian elimination etc. In order to achieve best results using this technique, the systems of equations must be overdefined as well as consist of linearly independent polynomials; else there is always a fair chance to get spurious results. Linearization technique can be very helpful specifically in case of block cipher's algebraic analysis because of the obvious reasons of degree swelling phenomena with each cipher round.

2.4.3 Relinearization

This techniques is an extension of linearization technique but is aimed at solution of an underdefined system of multivariate polynomial system. In such cases, we have more number of unknown variables but less number of equations. The core concept of relinearization technique is to add few extra non-linear equations after linearization, and then system is solved again by applying relinearization, until the system of equations is solved.

2.4.4 XL Algorithm

The XL or extended linearization algorithm was introduced in [7]. The concept behind this method is to generate more systems of equation with higher degree from any given systems of equations by multiplying given systems of equation with monomials of degree that is suitably selected. As a result, we get more number of equations than the variables but with higher degree. The resultant systems of equations use linearization technique for reducing the degree of equations for subsequent solution. It was shown [51] that XL attack method doesn't solves the problem in sub exponential time. Moreover, it also results in steep increase in memory requirement.

2.4.5 ElimLin

ElimLin stands for Eliminate Linear. So as the name indicate, this method searches for linear relations in the linear span of equations such that several variables are simply substituted by a linear expression. Consequently, new linear equations are obtained. This exercise of eliminating linear is repeated until no more linear equation are found, which means, no linear variables are found that can be eliminated. During this process, variables that are in smaller number are eliminated first, while variables with more quantity are eliminated at the last. Thus sparsity of the systems is maintained. The main idea of this algorithm is that the resultant systems of equations should always remain in degree 2. ElimLin algorithm has been used in algebraic cryptanalysis of reduced round DES [4].

2.4.5 SAT Solvers

SAT-solvers that have been traditionally used in industry for real world problems like software and hardware verifications based problems etc. SAT solvers; quite often, abbreviated as Satisfaibility Problem, have emerged as a most important and powerful tool in solution of complex systems of equations. Since the year 2000, every year a SAT race competition is held in order to make improvements in this technique. Accordingly, various versions of SAT solvers have been introduced including MinSAT and CyptoMinSAT. The basic idea is that once the cipher has been modelled into corresponding system of equations, which is in ANF form, the attacker then converts it into CNF form of boolean expression for finding out the possible solutions [52].

CNF is basically combinations of ANDs of ORs. The literals i.e. variables, and their negates are expressed in corresponding clauses containing logical ORs. These OR clauses are then combined using logical ANDs. CNF expression

consists of four different distinct heuristic. These includes, number of variables, number of clauses, average number of symbols per clause and total number of symbols in the system of CNF expressions. These CNF expressions are then fed to SAT solvers to find out the requisite solution.

SAT solvers are intrinsically heuristic based algorithm designed for solving SAT problems. The basic idea is to guess some of the variables then run the solver to examine the output. If a contraction is found then add a new clause. For enhancing the efficiency and reducing the complexity of overall problem, pre-processing of the systems of equations can give amazing results. Two approaches can be used in SAT based cryptanalysis. Details of the same are delineated below:

2.4.5.1 SAT Method

In this method, few X number of bits are guessed and solver is put to run. If assumption on X number of bits is correct, then SAT solver takes time T and gives the solution. Subsequently, solver operation can be aborted. The time complexity in this case is $2^X - T$. This method is commonly termed as SAT method.

2.4.5.2 UNSAT Method

Certain X number of bits can be guessed and SAT solver is put to the run mode. If assumption on X bits is incorrect; then there could be two possibilities. Firstly, the solver finds contradiction in time T with very large probability $1\text{-}P$ say 90% or it can find the contradiction with a small probability $P > 0$. If it finds contraction with small probability, few other bits also guessed to find additional contradiction or solution. Ultimately, if P is very small, the complexity of additional steps will be less than $2^X.T$ that is the time solver took during initial phase while finding contraction. This method is known as UNSAT method.

Nonetheless, in practical scenarios both SAT and UNSAT methods are employed and complexity of both the individual method is added towards the total complexity of the solver. SAT solvers are being used in algebraic cryptanalysis of block ciphers like DES [4], KEELOQ [5]. Moreover, it was shown in [49] that SAT solvers do solve complex system and they have very low memory consumption requirements.

2.5 Conclusion

In this chapter, an overview, design rationale and basic philosophy behind the advent of light weight block ciphers have been discussed. After describing the literature review regarding light weight block ciphers, the basic concept of algebraic cryptanalytic technique has been discussed. Later on, the fundamental methodology regarding algebraic techniques in block ciphers cryptanalysis has been discussed in light of published literature. Various tools and techniques available for solution of complex system of equations with particular reference to algebraic cryptanalysis of block ciphers have been discussed. Among these, traditional groebner basis approach based algorithms like F4 and F5 and their applicability to block ciphers cryptanalysis has been described. ElimLin algo, which tries to keep the equations system in 2^{nd} degree, can be an interesting tool as for as block cipher cryptanalysis is concerned. Finally, an emerging and very powerful technique, namely SAT solvers used in algebraic cryptanalysis of block ciphers has also been discussed.

Chapter 3

Algebraic Representation and Analysis of Light Weight Block Ciphers

3.1 Introduction

In this chapter, the methodology used in this thesis for formulating systems of equations and analysis of feistel structure based light weight block cipher has been discussed. Detailed framework for describing any block cipher into system of equations has been presented. A strong foundation, encompassing development of software tool, has been laid for usage in algebraic representation and analysis of targeted cipher in subsequent chapters.

Chapter 3 has been organized into 3 sections. Section 3.2 describes detailed methodology for producing algebraic representation of any block cipher. Initially, a step by step methodology for transforming any S-box into equivalent systems of linearly independent equations has been discussed that can completely describes pertinent S-box. Proposed methodology has been explained and implemented with the help of a prototype S-box. This lead to the development of a software tool that can analyze any lower order S-box. It has also been demonstrated that proposed methodology gives better results than previously reported. Section 3.3 contains another technique proposed for calculating estimated complexity of Linearization based i.e. eXtended Sparse Linearization (XSL) attack for block ciphers. XSL attack has also been discussed along with procedure to calculate its complexity. Section 3.4 concludes the chapter.

3.2 Modeling Light Weight Block Ciphers into Systems of Equations

Unlike stream ciphers, completely different methodology to transform block cipher into systems of equation has to be adopted. It is based on the broad outlines proposed by Cid et al. [13], Biruykov et al. [37] and Courtois et al. [44] for algebraic representation of AES and other ciphers based on feistel as well as SPN structure. Subsequent to the emergence of captioned methodology, algebraic representation and analysis of various block ciphers have been published including DES [4], AES [13], GHOST [35, 36, 53] and CTC [45, 46]. The concept is that instead of constructing extremely complex and large system of equations describing any cipher text bit relationship with plaintext and key bits, a simpler sparse polynomial systems of equations with lower degree can be obtained separately for linear and non-linear layers. They are then combined to form overall systems of equations. This methodology gives very large system of multivariate equations. Addition of intermediate state variables keeps the degree of equations systems lower at the cost of more variables. Moreover, encryptions of more number of data also don't ease the problem, since its gives more number of equations with more variables.

Base on the idea presented in [13, 37] in order to describes block cipher into systems of equations, the attacker formulates separate systems of equations for main cipher and key scheduling; separately for linear components and nonlinear components. These are then combined to formulate overall systems of equations. So the attacker starts with the nonlinear components; S-boxes, since security of any block cipher in general and against algebraic cryptanalysis in

particular lies in the S-box. Therefore, formulating equation systems representing the S-box are the trickiest part that determines the overall complexity of attack.

3.2.1 Step by Step Methodology for describing any S-box into Systems of Equations

It was shown in [37, 44, 54] that any S-box can be described into equivalent systems of equations. However, the detailed methodology has not been published. Doing exhaustive search even for small S-box over entire space of input/output is not practicable. In this section, a step by step methodology has been described that can be applied to any S-box for its algebraic representations involving linearly independent relations that holds good for all input/output of the pertinent S-box. The generated equations are usually both overdefined and sparse. The beauty of this technique is that relations involving any degree can be generated, however, since incase of iterative block cipher, there is fear of degree swell with each number of rounds, so generally it is desired to formulate and rely on second degree equations for the S-box. However, if the S-box is of higher I/O order and can't be described completely using MQ systems, one can always derive the 3^{rd} degree and so on equations as in case of DES [4]. For ease of assimilation and description, let's suppose simpler 3x3 S-box as in [54] with contents described in equivalent binary delineated in table 3.1. The proposed methodology for describing S-box into systems of equations used in this thesis has been presented in ensuing paragraph:

3.2.1.1 Define Monomials Based On Equations Degree Required to be Obtained - Step 1

For ease of description, methodology for generation of quadratic systems of equations has been described. Accordingly, the monomial defined are *"1",*"

"x_0", "x_1", "x_2", "x_1", "y_0", "y_1", "y_2", "$x_0\ y_0$", "$x_0\ y_1$", "$x_0\ y_2$", "$x_1\ y_0$", "$x_1\ y_1$", "$x_1\ y_2$", "$x_2\ y_0$", "$x_2\ y_1$", "$x_2\ y_2$". Moreover, all possible combinations of $y_i\ y_j$ can also be defined for generating additional number of equations. Similarly if the aim is to generate 3rd degree equations as well, so all possible combinations of monomials encompassing $x_i x_j y_k$ and $y_i y_j x_k$ can also be defined. But for easier assimilation and description and as a Proof of Concept (PoC) of the methodology, deliberately simpler monomials for captioned S-box have been defined.

Table 3.1 Contents of a 3 x 3 S-box

Input		Output	
Decimal	Binary	Decimal	Binary
0	000	5	101
1	001	3	011
2	010	0	000
3	011	4	100
4	100	2	010
5	101	7	111
6	110	6	110
7	111	1	001

3.2.1.2 Matrix Generation – Step 2

Next step is to generate a matrix of the order $M \times 2^n$; where M is total no of monomials and n is the input bits of the S –box. In this case 16x8 matrix is generated, as appended in table 3.2. The constant term is denoted by all 1s, whereas, other respective I/O monomials are designated with corresponding bits. The order of bit can be fixed either way like for example for 6 it can be either 110 or 011, depending upon the nomenclature followed for S-Box labeling (x_0, x_1, x_2

or x_2, x_1, x_0). Rests of the bits are simple bit by bit multiplication of the initial I/O variables values.

1st Input /output: Input 0 means binary 000----> output 5 means binary 101

2nd Input /output: Input 1 means binary 001----> output 3 means binary 011

And so on……..

8th Input /output: Input 7 means binary 111----> output 1 means binary 001

Table 3.2 Matrix Generation for 3 x 3 S-box

Monomial	1st I/O	2nd I/O	3rd I/O	4th I/O	5th I/O	6th I/O	7th I/O	8th I/O
1	1	1	1	1	1	1	1	1
x_0	0	0	0	0	1	1	1	1
x_1	0	0	1	1	0	0	1	1
x_2	0	1	0	1	0	1	0	1
y_0	1	0	0	1	1	0	1	0
y_1	0	1	0	0	1	1	1	0
y_2	1	1	0	0	1	0	0	1
$x_0 .y_0$	0	0	0	0	1	0	1	0
$x_0 .y_1$	0	0	0	0	1	1	1	0
$x_0 .y_2$	0	0	0	0	1	0	0	1
$x_1 .y_0$	0	0	0	1	0	0	1	0
$x_1.y_1$	0	0	0	0	0	0	1	0
$x_1.y_2$	0	0	0	0	0	0	0	1
$x_2 .y_0$	0	0	0	1	0	0	0	0
$x_2.y_1$	0	1	0	0	0	1	0	0
$x_2 .y_2$	0	1	0	0	0	0	0	1

Consequently, from above table, we get a 16x8 matrix, where each monomial designates an eight bit array (row of the matrix).

3.2.1.3 Gaussian Elimination of the Generated Matrix – Step 3

The matrix generated from above step is Gaussian eliminated mod 2. Resultantly, matrix in row echelon form is obtained, where few rows have non zero bits while all other have all zero bits. The number of all zero rows describes that how much equations will be generated for this specific S-box. From matrix in Table 3.3, there are 08 all zero rows, which indicates that in this case 08 equations of 2^{nd} degree shall be generated. Next steps are aimed at finding these equations.

Table 3.3 3 x 3 S-box Matrix in Reduced Row Echelon Form

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

3.2.1.4 XOR Mod 2 Monomials and Output Results – Step 4

In this step, all monomials are XORed modulo 2 in all possible combinations of 2 monomials, 3 monomials and so on till 6 monomials (because based on selected 3x3 S-box we have 7 basic monomials, 3 each for input and output and one constant). Consequently, corresponding 8 bit arrays are XORed bit by bit. The aim is to output all those combinations whose XOR gives all zero. Thus, ℓ number of equations from 2 up till 7 terms are generated. In this case,

total 100 equations are generated; having XOR output all zero. They are listed at Appendix A to this report.

3.2.1.5 Generate Matrix from the Equations – Step 5

Next step is to generate a matrix from the equations formulated in the step-4. Where the monomial is present that means it has co-efficient 1, else zero. So output of this step is *l x M* matrix; where each element in row describes the co-efficient of monomial in the particular equation, while one complete equation describes a row. It is important to note that matrix can be written from equation in either lexicographical order or reverse-lexicographical order ($1+x_1+x_1x_2$ *or* $x_1x_2+x_1+1$). It has been verified that both give valid results. In case of 3x3 S-box, a matrix of the order 100x16 is generated. However, even for 4x4 S-box, order of the matrix becomes extremely dense; few hundred thousand rows.

3.2.1.6 Gaussian Elimination Mode 2 of *l x M* Matrix – Step 6

In this step, Gaussian elimination Mod 2 on *l x M* matrix is performed. This gives exactly the number of non-zero rows that are equal to the number of all zero rows obtained in the resultant matrix that was obtained as result of step 3. Each row of this resultant matrix gives the linearly independent equations that describe the S-box. In this case while formulating the matrix in lexicographical order, following equations are obtained.

$$1+x_0+x_1+y_2+x_0y_0 = 0$$

$$x_0+x_0y_1+x_1y_2 = 0$$

$$x_2+y_1+x_0y_2+x_1y_0 =0$$

$$y_0+y_2+x_1y_0 +x_2y_2 = 0$$

$$y_1+x_0y_0+x_2y_1 = 0$$

$$x_0y_0+x_0y_1+x_1y_2+x_2y_1+x_2y_2 = 0$$

$$x_0y_1+x_0y_2+x_1y_0+x_2y_0+x_2y_1+x_2y_2 = 0$$

$$x_1y_0+x_1y_1+x_2y_0 = 0$$

3.2.1.7 Verification of Equations for all Input/Output of S-box – Step 7

Last but not the least, one must check that whether the equations generated through abovementioned process are valid for all input/output of the S-box. So a truth table is generated for verification. Conversely, $f(x_0, x_1, x_2, y_0, y_1, y_2)$mod2 should equate to zero for all the values of I/O bits of pertinent S-box . In this case it can be easily verified that these equations are valid for all I/O of the 3 x 3 S-box.

3.2.1.8 Some Findings Based on Analysis of Methodology for Formulating S-Box Equations

In this section, the analysis of above mentioned methodology has been outlined. As discussed earlier, the matrix in step 5 can be formulated by writing monomials coefficients either in lexicographical order or reverse-lexicographical order. While formulating the matrix in lexicographical order, the resultants equations are described in step 6, however, while formulating it in reverse lexicographical order, following equations are obtained:

$$x_2y_2+y_1+x_0 = 0$$

$$x_2y_1+x_0y_0+y_1 = 0$$

$$x_2y_0+x_1y_1+x_1y_0 = 0$$

$$x_1y_2+x_0y_1+x_0 = 0$$

$$x_1y_1+x_0y_2+x_0y_1+x_0y_0+x_0 = 0$$

$$x_1y_0+y_2+y_1+y_0+x_0 = 0$$

$$x_0y_2+y_2+y_0+x_2+x_0 = 0$$

$$x_0y_0+y_2+x_1+x_0+1 = 0$$

Statistical analysis of the resultant equations in both the order is given in table 3.4 below. From statistical analysis of monomials in the S-box equations, it can be deduced that while formulating the matrix in step 5, reverse lexicographical order gives simpler and sparse systems of equations. This is due to the exploitation of property of Gaussian elimination because on the left side of reduced row echelon form of the matrix, all zeros are obtained. So it can be concluded that reverse lexicographical ordering is the preferred methodology for formulating simpler S-box equations.

Table 3.4 Statistical Analysis of Monomials in S-Box Equations

Statistical Analysis of Monomials - Lexicographical Order

Monomials	Count	Percent
x1y0	4	12.12%
x0y0	3	9.09%
x0y1	3	9.09%
x2y2	3	9.09%
x2y1	3	9.09%
x1y2	2	6.06%
y2	2	6.06%
x0y2	2	6.06%
x0	2	6.06%
y1	2	6.06%
x2y0	2	6.06%
x2	1	3.03%
x1y1	1	3.03%
y0	1	3.03%
x1	1	3.03%
1	1	3.03%

Statistical Analysis of Monomials – Reverse Lexicographical Order

Monomials	Count	Percent
x0	6	18.75%
y2	3	9.38%
x0y0	3	9.38%
y1	3	9.38%
x0y1	2	6.25%
x0y2	2	6.25%
x1y0	2	6.25%
x1y1	2	6.25%
y0	2	6.25%
x2	1	3.13%
x2y0	1	3.13%
x2y1	1	3.13%
x2y2	1	3.13%
x1	1	3.13%
1	1	3.13%
x1y2	1	3.13%

SUMMARY	
Total Monomials	33
Single Degree Monomials	9
2nd Degree Monomials	23
Constants	1

SUMMARY	
Total Monomials	32
Single Degree Monomials	16
2nd Degree Monomials	15
Constants	1

Moreover, in order to make algebraic attacks against any block cipher impracticable, it is imperative that at least few S-boxes in the cipher should not be described by small number of multivariate equations.

3.2.1.9 Comparison of Results Obtained with Published Results

Comparison of the results obtained from abovementioned step by step methodology using reverse lexicographical ordering with the already published results for the same S-box in [54], is appended in table 3.5 below:

Table 3.5 Comparison of Results for S-box Equation - Monomials

Title	Proposed Methodology	Previous Results
No. of Equations	8	8
No of Monomials	32	41
1st Degree Monomials	16	26
2nd Degree Monomials	15	11
Constants	1	4

From above comparisons, it can be deduced that the proposed methodology gives simpler and sparse systems of equations that completely

describes the S-box. Moreover, if a cipher like SEA [15] uses 8 S-boxes per rounds; each for key scheduling and round function, so for 32 rounds, proposed methodology gives 32*16*9 = 4608 less monomials in the entire span of equations, which is considerable reduction in complexity.

3.2.2 Equations systems for Linear Components

After the equations systems for nonlinear component in the cipher and key algorithm have been obtained, next step is to formulate equation systems for linear layers; separately for the cipher and key scheduling [13, 37]. Linear layer consists of diffusions layer as well key addition layer; prior or after the S-box layer and in-between two nonlinear layers. However, linear layers containing simple bit permutation and/or shift left/shift right function are not considered as separate equations, because they can be catered by simply re-ordering the variables/equations. Aim is to make the overall equation systems as sparse as possible while preserving the overdefindness.

3.2.3 Obtaining Equations Systems for the Complete Cipher Scheme

Once the equation systems for linear and nonlinear components; separately for cipher and key scheduling part, with certain number of variables and terms have been obtained, they all can be combined to form an overall systems of equations that describes the cipher as shown in [13, 37]. Intermediate state variables are added at each round to make the equations systems sparse. Resultantly, equations systems comprising certain number of variables and terms (or monomials) can be obtained, which gives the theoretical complexity of algebraic attacks, as in case of AES [13].

3.3 Complexity of XSL Attack

Shamir et al. in [7] showed that MQ problem is generally NP hard problem. However, drastic reduction in its complexity can be observed if it becomes overdefined. Based on same idea, a theoretical variant of algebraic attack known as eXtended Sparse Linearization (XSL) attack was proposed by Courtois et al. in [44]. It was proposed to exploit the sparsity and structure of the AES and Serpent equations system. XSL attack was primarily presented against ciphers that have pertinent structure encompassing key XOR, substitution layer followed by linear diffusion layer; however, it was shown in [12, 38, 44] that it can be applied to other block ciphers including feistel ciphers.

Since XSL is a variant of generic XL attack. So the basic difference between two is that in case of XL attack, each equation system is multiplied by all possible monomial of any degree *D-2*. Whereas, in case of XSL attack, more equations systems are generated by multiplying a suitably selected monomials. The monomial is chosen such that its product already appears in other equations. Thus, once its linearised, resultant systems becomes sparse. Courtois.et.al also gave a following generalized formula [44] that describes the estimated complexity of the work factor for XSL attack against block ciphers without taking into account the key scheduling:

$$W.F \cong \Gamma^{\omega}\,[(\text{Block Size}).\,(\text{Number of Rounds})^{2}]^{\omega\lceil t/r\rceil}$$

Where

$\Gamma = (t/s)^{\lceil t/r\rceil}$ = Actual contribution of S-box towards complexity of XSL attack

t = Number of monomials in S-box equations

r = Number of equations describing S-box

s = Input bits of S-box

ω = Constant describing Gaussian Elimination complexity and its value is 2.37

Though the performance of XSL attack against AES, as claimed in [44] has been debatable among the cryptographic community [12, 55] since the advent of this attack, however, its theoretical complexity against other ciphers like SPN based Serpent [XSL] or feistel based MIBS [56] has not been questioned in the published literature. Moreover, in [12], it was shown that even in its current form, the XSL method against AES is better than exhaustive search. Therefore, in order to analyze cipher resistivity against algebraic attacks the XSL formula gives the valid approximation of estimated complexity.

3.4 Conclusion

In this chapter, the adopted methodology for analysis of feistel structure based light weight block ciphers has been discussed. In the start, a step by step procedure for formulation of systems of equations for any S-box has been discussed with the help of an example. Based on the proposed methodology, an interesting finding for generating sparse and simpler system of equations has been explored and demonstrated. Then through adopted methodology, an improvement in results as compared to the previously published results has been presented. Later on the overall methodology, used for describing complete block ciphers into system of equations has also been described. At the end, XSL attack and the procedure for calculating its complexity has been discussed.

Chapter 4

Algebraic Representation and Analysis of LBlock Cipher

4.1 Introduction

In this chapter, algebraic representation and analysis of feistel structure based light weight block cipher has been presented. The chapter has been divided into 5 sections. Section 4.2 present structure of LBlock. Section 4.3 contain results and analysis of algebraic equations derived over LBlock S-boxes. Section 4.4 describes results of the algebraic representation of LBlock over complete 32 rounds. Section 4.5 calculates and analyses the resistivity of LBlock against XSL attack. Section 4.6 concludes the chapter.

4.2 LBlock Light Weight Block Cipher

LBlock is ultra light weight block cipher proposed in 2011[14]. Its design and development was sponsored by Chinese Academy of Science (CAS). The name LBlock emerged from LuBanlock and LightweightBlock cipher, shown in Figure 4.1

Figure 4.1 Luban Lock

LBlock has very efficient hardware implementation of 1320 Gate Equivalents (GE) on 0.18µm technology with a throughput of *200Kbps* at *100KHz*. Additionally; its software implementation on 8 bit microcontroller has also been proposed. Thus, making it an attractive candidate for employment in extremely resource constraints applications. LBlock has been extensively analyzed against Cube, Linear, Differential, Boomerang, Related Key and Biclique cryptanalysis [26, 27, 28, 29, 30]. However, its algebraic representation and analysis is yet to be reported. General specifications of LBlock are summarized below:

Table 4.1 General specifications of LBlock

Block Size	64 Bit
Key Size	80 Bit master key 32 Bit round key
Number of rounds	32
Non-linear components	4x4 S-box Eight S-boxes operating in parallel in cipher algorithm Two S-boxes in key scheduling
Linear components	Bit wise XOR operation Bit shift left and shift right operation

4.2.1 Structure of LBlock Encryption / Decryption Algorithm

LBlock's encryption algorithm consists of 32 round based on feistel structure. Each round takes 64 bit plaintext as an input and breaks it into two halves; 32 bit each. Thus, only 32 bit block size is processed in each round like typical feistel cipher. Encryption function of the algorithm is depicted below in Fig 4.2:

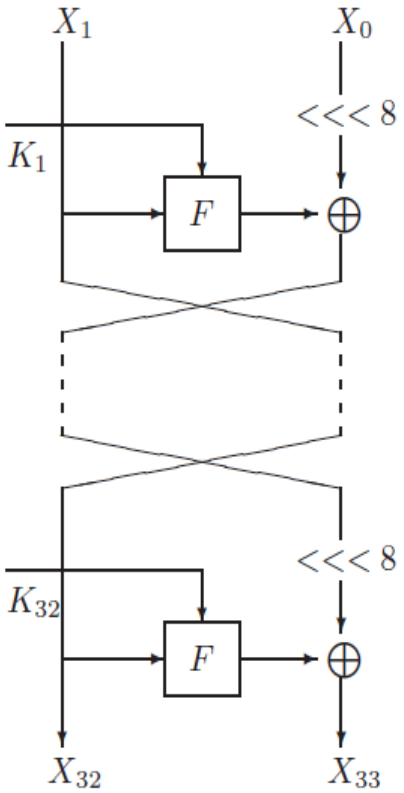

Fig 4.2 Encryption Function of LBlock

In each round, right half of 64 bit input text is passed through 8 bit shift left function, while other half is passed through F-function. These both bit blocks are then XORed to generate the left 32 bits output of round. While the right 32 bits input goes unprocessed to form the right half of the round's output. Round function F is described in Fig 4.3

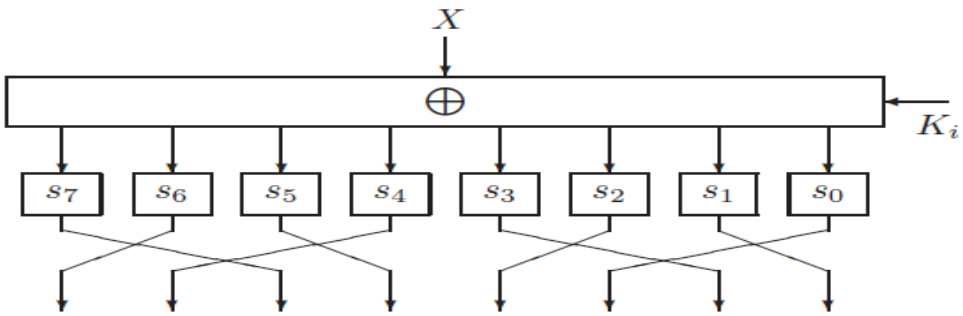

Fig 4.3 F-Function of LBlock

In round function, 32 bit text is XORed with 32 bit round key. Then resultant 32 bit is processed through eight, 4x4 bit S-boxes; thus generating confusion. The output of each S-box is permutated in the form of 4-bit word to generate diffusion effects. Contents of Sboxs used in LBlock are given in Table 4.2. Like other feistel schemes, the decryption function is similar to encryption function but operates in reverse order.

Table 4.2 Contents of LBlock S-boxes

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
s_0	14	9	15	0	13	4	10	11	1	2	8	3	7	6	12	5
s_1	4	11	14	9	15	13	0	10	7	12	5	6	2	8	1	3
s_2	1	14	7	12	15	13	0	6	11	5	9	3	2	4	8	10
s_3	7	6	8	11	0	15	3	14	9	10	12	13	5	2	4	1
s_4	14	5	15	0	7	2	12	13	1	8	4	9	11	10	6	3
s_5	2	13	11	12	15	14	0	9	7	10	6	3	1	8	4	5
s_6	11	9	4	14	0	15	10	13	6	12	5	7	3	8	1	2
s_7	13	10	15	0	14	4	9	11	2	1	8	3	7	5	12	6
s_8	8	7	14	5	15	13	0	6	11	12	9	10	2	4	1	3
s_9	11	5	15	0	7	2	9	13	4	8	1	12	14	10	3	6

4.2.2 Key Scheduling

Like traditional light ciphers, key scheduling of LBlock has been designed in a stream ciphers way. For strengthening cipher's resistivity against algebraic attacks, two 4x4 S-boxes have also been added in each round of the key scheduling part. Overview of the key scheduling is described in Fig 4.4.

Master Key K = 80 bits $\rightarrow$ 32 bit round key $K_{i\ (for\ i=1,\ 2,\ 3\ldots\ 32)}$

Obtain 32 leftmost bits from master key and proceed as follow:

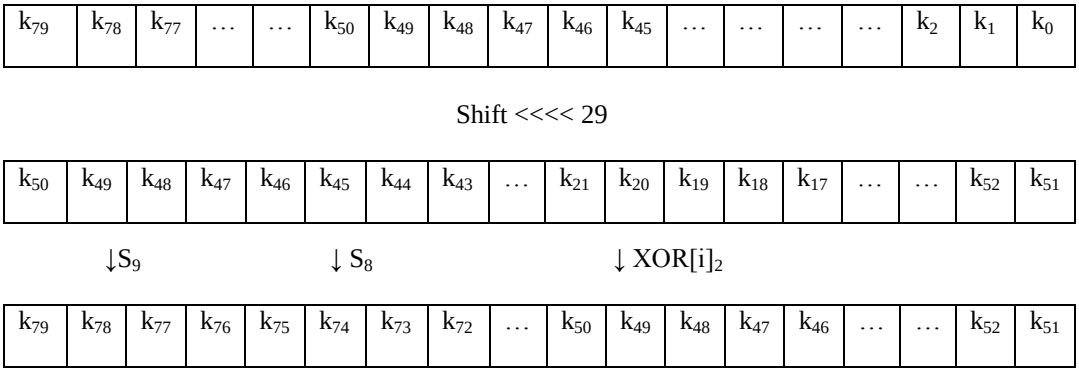

Fig 4.4 Key Scheduling of LBlock

4.3 Results of Algebraic Equations for S Boxes

Step by step methodology for transforming LBlocks Sboxs into systems of quadratic equations has been adapted using the developed tool. LBlock have 08, 4x4 S-boxes in encryption/decryption algorithm and 02, 4x4 S-boxes in key schedule. So in this case 37 monomials encompassing all I/O variables and their products are considered for formulating overdefined systems of equations that completely describes LBlock S-boxes. These includes; *"1", "x0", "x1", "x2", "x3", "y0", "y1", "y2", "y3", "x0y0", "x0y1", "x0y2", "x0y3", "x1y0", "x1y1", "x1y2", "x1y3", "x2y0", "x2y1", "x2y2", "x2y3", "x3y0", "x3y1", "x3y2", "x3y3", "x0x1", "x0x2", "x0x3", "x1x2", "x1x3", "x2x3", "y0y1", "y0y2", "y0y3", "y1y2", "y1y3", "y2y3"*. Then following the procedure as outlined in previous chapter, 21 equations for each S-box have been obtained. Equations for S0 of LBlock are appended below, while equations systems for remaining 09 S-boxes are placed at Appendix B.

$$y2y3 + y1y3 + x2y3 = 0$$
$$y1y3 + y1y2 + x2y1 = 0$$
$$y1y2 + x3y1 + x2y1 + y1 = 0$$
$$y0y3 + x3y0 + y0 = 0$$

$$y0y2 + x2y2 + x1y2 + x0y2=0$$

$$y0y1 + x0x3 + x3y1 + x1y0=0$$

$$x2x3 + x1x2 + x2y1=0$$

$$x1x3 + x3y3 + x0y3 + x0y0=0$$

$$x1x2 + x3y2 + x2y1=0$$

$$x0x3 + x3y1 + x3y0 + x2y0 + x1y0 + x0y0 + y0=0$$

$$x0x2 + x3y2 + x2y1 + x2y0 + x2=0$$

$$x0x1 + x3y2 + x2y1 + x2y0 + x1y3 + x0y2 + x2 + x1=0$$

$$x3y3 + x3y1 + x2y0 + x1y0 + x0y3 + y0 + x3=0$$

$$x3y2 + x3y1 + x2y1 + x2y0 + x1y3 + x1y2 + x1y0 + x0y3 + y0 + x3=0$$

$$x3y1 + y2 + y1 + x2=0$$

$$x3y0 + x2y3 + x2y2 + x2y0 + x1y3 + x1y2 + x1y1 + x1y0 + x0y3 + x0y0 + y2 + y1 + x3=0$$

$$x2y3 + x1y3 + x1y2 + x1y0 + x0y3 + x0y1 + y0 + x3=0$$

$$x2y2 + x2y0 + x1y2 + x1y0 + x0y3 + x0y2 + x0y0=0$$

$$x2y1 + x2y0 + x1y3 + x1y2 + x1y0 + x0y3 + y2 + y1 + x1 + x0=0$$

$$x2y0 + x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y0 + x3=0$$

$$x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y1 + y0 + x0 + 1=0$$

Summary of all S-boxes equations derived over LBlock is appended below in table 4.3:

Table 4.3 Summary of LBlock Sbox Equations - Degree of Monomials

S-box in Encryption/Decryption Algorithm				
S-box	No of Equations	1st Degree Monomials	2nd Degree Monomials	Constants
S0	21	31	103	1
S1	21	38	85	4
S2	21	45	88	5
S3	21	37	88	3
S4	21	41	83	2
S5	21	41	80	5
S6	21	44	71	1
S7	21	33	98	1

Total	**168**	**310**	**696**	**22**
S-box in Key Scheduling				
S8	21	32	85	1
S9	21	29	95	1
Total	**42**	**61**	**180**	**2**

All equations have been duly verified for all I/O variable of pertinent S-box using *f(x0, x1, x2, x3, y0, y1, y2, y3)* function in Maple for all $x_{i,j}$. Statistical analyses of individual S-box equations for all LBlock S-boxes are placed Appendix C. While statistical analysis of individual S-box equations with respect to I/O variables is placed at Table 4.4.

Table 4.4 Statistical Analysis of S-box Equations – I/O Degree of Variables

SBOX0

Variable	Count	Percent
y0	36	15.13%
x1	32	13.45%
x0	29	12.18%
x2	29	12.18%
y2	29	12.18%
y1	28	11.76%
y3	25	10.50%
x3	19	7.98%
1	1	0.42%

SBOX1

Variable	Count	Percent
x0	34	16.04%
y1	31	14.62%
x2	28	13.21%
y0	25	11.79%
y2	24	11.32%
x1	20	9.43%
x3	20	9.43%
y3	17	8.02%
1	4	1.89%

SBOX2

Variable	Count	Percent
x2	33	14.60%
x0	31	13.72%
y3	31	13.72%
y1	28	12.39%
x3	27	11.95%
y0	26	11.50%
y2	20	8.85%
x1	15	6.64%
1	5	2.21%

SBOX3

Variable	Count	Percent
y2	31	14.35%
x0	28	12.96%
x2	28	12.96%
y0	28	12.96%
x1	24	11.11%
x3	24	11.11%
y1	18	8.33%
y3	18	8.33%
1	3	1.39%

SBOX4

Variable	Count	Percent
x0	34	16.27%
y2	27	12.92%
y1	26	12.44%
x3	25	11.96%
y0	24	11.48%
x1	23	11.00%
x2	19	9.09%
y3	19	9.09%
1	2	0.96%

SBOX5

Variable	Count	Percent
x0	29	14.08%
y0	29	14.08%
y2	29	14.08%
x2	28	13.59%
x3	21	10.19%
y1	21	10.19%
x1	17	8.25%
y3	17	8.25%
1	5	2.43%

SBOX6

Variable	Count	Percent
x0	29	15.51%
y0	29	15.51%
x3	25	13.37%
y1	22	11.76%
x2	21	11.23%
y2	19	10.16%
x1	15	8.02%
y3	15	8.02%
1	1	0.53%

SBOX7

Variable	Count	Percent
y1	36	14.88%
x1	32	13.22%
x0	31	12.81%
y0	29	11.98%
x2	28	11.57%
y2	26	10.74%
y3	25	10.33%
x3	23	9.50%
1	1	0.41%

SBOX8

Variable	Count	Percent
x0	35	17.07%
y1	30	14.63%
x2	27	13.17%
y0	25	12.20%
x1	21	10.24%
y3	21	10.24%
x3	18	8.78%
y2	17	8.29%
1	3	1.46%

SBOX9

Variable	Count	Percent
x2	30	13.64%
x1	29	13.18%
y1	29	13.18%
y2	28	12.73%
y0	26	11.82%
y3	25	11.36%
x0	24	10.91%
x3	18	8.18%
1	1	0.45%

4.3.1 Some Findings on LBlock S-Box Equation

Each S-box of LBlock can be completely defined in terms of 21 MQ system of equations. Since each S-box has 4 I/O variables, so 21 equations in terms of 8 variables are obtained which is an overdefined system. To reduce the complexity of attack, one may choose simpler set of equations among these 21 systems of equations.

Based on statistical analysis of individual S-box equations with respect to I/O variables as presented in table 4.4, it is evident that at least one input of S-box occurs with more number than others, so if these can be determined through CP or any other technique, then the complexity of the overall systems of equation over LBlock and associated algebraic attack can considerably reduce.

4.4 Constructing Equations Systems over Complete LBlock

As discussed in the previous chapter and based on [13, 37], equations systems over individual components; separately for cipher and key scheduling part, are combined to form the systems of equations that completely describe the ciphers. Complexity of the overall systems of equations that complexly describes

the 32 round LBlock over GF(2) is at table 4.5, while contribution of each individual component has been described in following subsections.

4.4.1 Equations Systems over LBlock Encryption/Decryption Algorithm

These include variables, nonlinear equations, and linear equations over the encryption/decryption algorithm of LBlock. Variables for the encryption algorithm are chosen to be the input/output of the S-boxes in each round to make the system of equations sparse while keeping them in degree 2 despite increase in number of rounds. Only nonlinear components in LBlock cipher algorithm are S-boxes. LBlock uses 8 S-boxes, each having 4 bit I/O, therefore, *32*8*8=2048* variables over complete 32 rounds of LBlock encryption/decryption algorithm are contributed by the substitution layer.

Moreover, 21 MQ systems completely define each LBlock S-box. So in totality, substitution layer in cipher algorithm contributes 8*8*32=5376 equations with 310*32=9920 1^{st} degree monomials, 699*32=22,368 2^{nd} degree monomials and 22*32=704 constant monomials. Similarly the each linear layer of key XOR prior substitution layer and plaintext XOR after substation layer contributes 32 linear equations with 64 first degree monomials or constants. So they add 2*32*32= 2048 equations with 2*2048=4096 1^{st} degree including constant monomials. Shift left function prior XOR and permutation layer after substitution don't add any additional equations, since they are adjusted through simple re-ordering.

Coming over the key scheduling part, only nonlinear component in LBlock key scheduling is substitution layer encompassing two 4 I/O bit S-boxes. Variables in the key scheduling part are 80 bit input key and 8 variables per S-box,

so overall key scheduling adds 80+(8*2*32)=580 variables into LBlock equations system. Nonlinear equations are also generated by S-boxes, so they contribute 21*2*32=1344 MQ equations, with 32*21=672 1^{st} degree monomials, 180*32=5760 2^{nd} monomials and 32*2=64 constant terms. Linear part of cyclic shift left and right in the key scheduling don't add any additional equations since they are adjusted through simple re-ordering of the bits, moreover, only linear layer that adds linear equations is XOR with round number. Therefore, it's contribution towards overall system is 32*5=160 equations with 160*2=320 1^{st} degree monomials.

Table 4.5 Systems of Equations for LBlock over GF (2)

Cipher	Variables	2048
	Non-linear Layer Equations	5376
	1^{st} Degree Monomials	9920
	2^{nd} Degree Monomials	22,368
	Constants Terms	704
	Linear Layer Equations	2048
	1^{st} degree Monomials	4068
	Constant Terms	32
Key Scheduling	Variables	580
	Non-linear Layer Equations	1344
	1^{st} Degree Monomials	672
	2^{nd} Degree Monomials	5760
	Constants Terms	64
	Linear Layer Equations	160
	1^{st} degree Monomials	320
MQ System for Complete LBlock	Variables	2628
	Equations	8928
	Monomials (Terms)	43,908
	2^{nd} Degree Monomials	28,128
	1^{st} Degree Including Constants	15,780

4.5 Complexity of XSL Attack over LBlock

According to [44], estimated complexity of XSL attack can be calculated against any block cipher. Thus, in case of LBlock estimated Work Factor (W.F) for complexity of XSL attack with block size of 64 bit, key size 80 bit, no of rounds 32, no of equations per S-box 21 with 37 variables shall be as follows:

$$\Gamma = (37/4)^{\lfloor 37/21 \rfloor} = 85.56 \cong 2^6$$

$$W.F \cong (2^6)^{2.37}[(64).(32)^2]^{2.37^{\lfloor 37/21 \rfloor}}$$

$$= 2^{14.22}[(2^6).(2^5)^2]^{2.37.2}$$

$$= 2^{14.22}[2^{16}]^{4.74}$$

$$= 2^{90}$$

Which indicates that the complexity of XSL attack is much higher than the exhaustive search of 2^{80} operations. Based on this fact, it can be concluded that LBlock is pretty secure against XSL attack.

4.6 Conclusion

In this chapter, each S-box of LBlock has been completely described in system comprising 21 MQ equations. Then, algebraic representation of complete 32 rounds of LBlock in terms of 2628 variables, 8928 equations and 43,908 terms has been presented. Since the best known algebraic attack against block cipher is against 6 round DES [4] through solving systems involving 2900 variables, 3030 equations in 4331 monomials, therefore, LBlock system is more complex than reduced round DES. Consequently, it can be concluded that practical algebraic attack against LBlock is infeasible. Moreover, even if XSL attack works as claimed [44], LBlock is adequately secure against it.

Chapter 5

Algebraic Representation and Analysis of Scalable Encryption Algorithm

5.1 Introduction

In this chapter, algebraic representation and analysis of feistel structure based light weight Scalable Encryption Algorithm (SEA) has been presented. The chapter has been organized into 5 sections. Section 5.2 presents structure of SEA. Section 5.3 contains results and analysis of equations derived over Substitution layer of SEA. Section 5.4 presents algebraic representation and analysis of two variants of SEA over GF (2). Section 5.5 calculates and analyses the resistivity of SEA against XSL attack. In the end, section 5.6 concludes the chapter.

5.2 Scalable Encryption Algorithm for Resource Constrained Applications

Scalable Encryption Algorithm ($SEA_{n, b}$) is a light weight block cipher based on feistel structure that was proposed in [15] for extremely resource constrained applications. Its design was sponsored by National Scientific Research, Belgium. Generally, the design of light weight ciphers are driven by the three basic factors; cost, security and performance. However, this traditional approach has not been followed for designing SEA. Instead, the core focused was on the kind of targeted applications that have extremely limited processing resources and subsequently very low throughput requirements. The important feature of this design is that it has exclusively been designed for implementation

on those kind of processors that have very limited instructional set; i.e. XOR, OR and AND gates, bit/word rotation and modular addition etc. Another interesting feature of its design is that SEA has very flexible text, key as well as processor size. Therefore, all these parameter can be chosen to suit the requisite demand of application.

Although, it can be implemented on any platform with flexible processor size, but its implementation on 8-bit microcontroller makes SEA a very strong candidate and popular choice for tiny implementation with extremely low computational power coupled with low throughput requirements. Moreover, recently, its FPGA implementation [31, 32], performance improvement in terms of implementation [33], low energy implementation [34] for light weight embedded applications has also been published. SEA has been extensively analyzed by the designers with respect to linear and differential cryptanalysis, truncated differential and square attacks [15], however, its algebraic representation and analysis is yet to be undertaken. General Specifications of SEA are enlisted in Table 5.1:

Table 5.1 General Specifications of SEA

Plain Text, Block Size and Key Size (n)	Flexible (The only limitation is that it has to be multiple of $6b$)
Processor Size or Word Size (b)	Processor size can be any, like 8 bit.
Number of rounds (n_r)	Flexible but odd $(3n/4 +2)* \lfloor n_b+b/2 \rfloor$
Structure	Feistel
Number of words per feistel branch (n_b)	$n / 2b$
Non-linear components	3x3 S-boxes operating in parallel; both in key scheduling and encryption

	algorithm
Linear components	Bit wise XOR operation Bit rotation (*r*) Word rotation (R and R^{-1}) Modular addition (mod 2^b)

5.2.1 Structure of SEA Encryption/Decryption Algorithm

Design of SEA is based on feistel structure. Like any typical feistel structure based cipher, only half of the block size is processed in each round. Moreover, to facilitate economical and easy implementation, it uses simple mathematical operation of 3 bit substitution, operating in parallel on input data to provide confusion, word rotation *R*, inverse word rotation R^{-1}, and bit rotation r to provide diffusion. Another operation used is mod 2^b addition of round key and half of input data block. This improves diffusion as well nonlinearity while safeguarding the cipher against different sorts of structural attacks [15]. The structure of encryption/decryption algorithm is described in Fig 5.1.

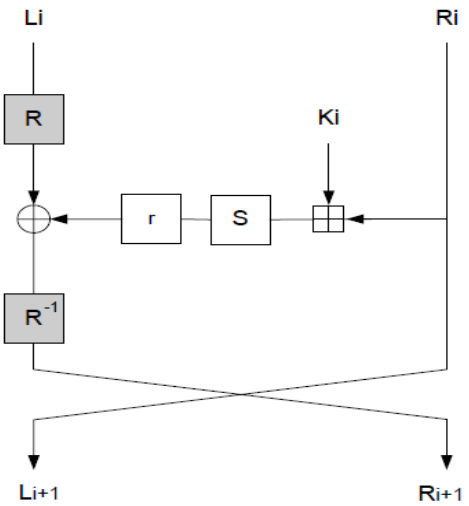

Fig 5.1 Encryption/Decryption Algorithm of SEA

The contents of 3 x 3 Box used in SEA are also delineated in table 5.2.

Table 5.2 Contents of S-box in SEA

Input	Output
0	0
1	5
2	6
3	7
4	4
5	3
6	1
7	2

5.2.2 Structure of SEA Key Scheduling Algorithm

Like the encryption/decryption algorithm, SEA follows similar feistel structure for the key scheduling part as well. Key scheduling uses, 3x3 S-boxes in parallel, word rotation *R*, bit rotation *r* and modular addition mod 2^b with C_i, Where C_i is an n_b word vector having all zeros bits except Least Significant Word (LSW); whose values is updated as per the number of round. Overall structure of key scheduling algorithm is depicted below in fig 5.2:

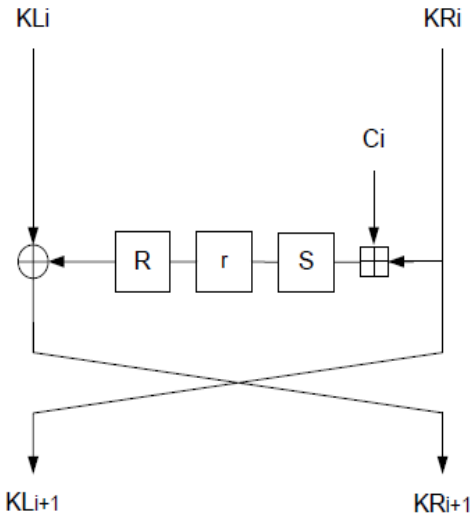

Fig 5.2 Key Scheduling Algorithm of SEA

5.3 Results of Algebraic Equations for S Box

SEA uses substation layer consisting of 3x3 S-box, operating in parallel with exact number of S-boxes depending upon the block size of data chosen. Since the S-box has 3 I/O variables, so using the developed tool, monomials encompassing I/O variables and their all possible combinations are used to generate overdefined systems of equations. Precisely, these are *"1", "x0", "x1", "x2", "y0", "y1", "y2", "x0y0", "x0y1", "x0y2", "x1y0", "x1y1", "x1y2", "x2y0", "x2y1", "x2y2", "x0x1", "x0x2", "x1x2", "y0y1", "y0y2", "y1y2"*. After following the step by step procedure, following 14 quadratic systems of equations for SEA S-box are obtained:

$$y1y2+x1y2=0$$
$$y0y2+x0y2=0$$
$$y0y1+x0x2+x0x1=0$$
$$x1x2+x0x2+x2y0=0$$
$$x0x2+x2y1=0$$
$$x0x1+x2y1+x2y0+x1y0=0$$
$$x2y2+x1y0+x0y1+x2=0$$
$$x2y1+x1y2+y1=0$$
$$x2y0+x0y1+x2=0$$
$$x1y2+x0y2+y1+x0=0$$
$$x1y1+x1y0+x0y1+y1+y0+x0=0$$
$$x1y0+x0y2+x0y1+x0y0+y0+x0=0$$
$$x0y2+y1+y0+x1=0$$
$$x0y1+x0y0+y2+y1+y0+x2+x0=0$$

All these equations have been verified for all I/O of the SEA S-box. Statistical analysis of SEA S-box equations With Respect To (WRT) individual monomials and I/O variable is appended below in table 5.3.

Table 5.3 Statistical Analysis of SEA S-box

W.T.T Individual Monomial

Monomials	Count	Percent
y1	5	9.43%
x0y1	5	9.43%
x0	4	7.55%
x0y2	4	7.55%
y0	4	7.55%
x1y0	4	7.55%
x0x2	3	5.66%
x2y1	3	5.66%
x1y2	3	5.66%
x2y0	3	5.66%
x2	3	5.66%
x0x1	2	3.77%
x0y0	2	3.77%
x1y1	1	1.89%
x1x2	1	1.89%
x2y2	1	1.89%
x1	1	1.89%
y0y1	1	1.89%
y0y2	1	1.89%
y1y2	1	1.89%
y2	1	1.89%

W.R.T I/O Variables

word	count	percent
x0	19	21.59%
y1	15	17.05%
y0	14	15.91%
x1	11	12.50%
x2	11	12.50%
y2	11	12.50%

Summary of Monomials

No of Equations	14
1^{st} Degree	18
2^{nd} Degree	35

5.3.1 Some Findings over SEA S-box Equations

14 systems of equations in 6 variables have been formulated for SEA S-box, which is an overdefined system of equations. However, simpler and fewer systems of equations from the set of 14 equations can be chosen for SEA S-box to reduce the overall complexity of attack. Moreover, from statistical analysis of I/O variables it is evident that first input to the S-box comes in very higher number in

the resultant systems of equations, so if this first bit of each s-box can be determined through some means than the overall systems of equations can become very simplified and sparse, thus drastically reducing the attack complexity.

5.4 Equations Systems over Complete Cipher

First equations systems over $SEA_{48,8}$ for 33 rounds have been constructed. That means that it has a key, PT and CT 48 bit, eight 3x3 s-boxes in main cipher algorithm and key schedule. First the variables and equations over cipher algorithm are described. In this case, variables are input/output of each S-boxes which means 6*8*33=1584. Each s-box has been completely described into 14 systems of equations over 6 unknown. Therefore, contribution of substitution layer toward overall system of equations is 14*8*33=3696 equations, having 18*8*33=4752 1st degree monomials and 35*8*33=9240 second degree monomials. The linear layer of word rotation, inverse word rotation and bit rotation doesn't add any additional equations since they are described through simple re-ordering of bits and words. Therefore, contribution of linear layer addition mod 2^b prior the substitution layer and XOR after the substation layer gives 48*33=1584 linear equations in 1584*2=3168 linear terms.

In key scheduling, $SEA_{48,8}$ also have 8 S-boxes each having 3x3 I/O variables. Thus contribution of $SEA_{48,8}$ key scheduling algorithm in terms of variables is I/O of S-boxes plus the input key variables. These came out to be (6*8*33) + 48=1632. Again the nonlinear component in key scheduling are S-boxes and their contribution to the systems of equations is 14*8*33=3696 in 18*8*33=4752 1st degree and 35*8*33=9240 second degree monomials. Linear layer of bit rotation and word rotation don't contribute to the overall system of

equations since they are adjusted by re-ordering the respective bits and words. The only nonlinear layer that contributes to the overall system of equations is XOR after and mod 2^b addition prior the substation layer. Their contribution is 33*48=1548 equations in 3168 terms. Similarly, the equations systems for $SEA_{96,8}$ over 33 rounds have also been constructed, summary of the same is given below in table 5.4.

Table 5.4 Systems of Equations for SEA over GF (2)

SEA Version		$SEA_{48,8}$	$SEA_{96,8}$
Cipher	Variables	1584	3168
	Non-linear Layer Equations	3696	7392
	1^{st} Degree Monomials	4752	9504
	2^{nd} Degree Monomials	9240	18,480
	Linear Layer Equations	1584	3168
	1^{st} degree Monomials	3168	6336
Key Scheduling	Variables	1632	3264
	Non-linear Layer Equations	3696	7392
	1^{st} Degree Monomials	4752	9504
	2^{nd} Degree Monomials	9240	18480
	Linear Layer Equations	1584	3168
	1^{st} degree Monomials	3168	6336
MQ System for Complete SEA	Variables	3216	6432
	Equations	10,560	21,120
	Monomials (Terms)	34,320	68,568
	2^{nd} Degree Monomials	18,480	36,960
	1^{st} Degree Constants	15,840	31,608

5.5 Complexity of XSL Attack over SEA

According to [44], estimated W.F. for complexity of XSL attack can be calculated against any block cipher. Thus, in case of $SEA_{48,8}$ estimated W.F. for

complexity of XSL attack with block and key size of 48 bit, No. of rounds 33, No. of equations per S-box 14 with 22 initial variables shall be as follows:

$$\Gamma = (22/3)^{\lceil 22/14 \rceil} = 53.88 \cong 2^6$$

$$W.F \cong (2^6)^{2.37}\,[(48).(33)^2]^{2.37^{\lceil 22/14 \rceil}}$$

$$= 2^{14.22}[(2^5).(2^5)^2]^{2.37.2}$$

$$= 2^{14.22}[2^{15}]^{4.74}$$

$$= 2^{85.2}$$

Which indicates that the complexity of XSL attack is much higher than the exhaustive search of 2^{48} operations. Similarly for $SEA_{96,8}$, the estimated complexity of XSL attack comes out to be around $2^{94.8}$, which is again not very less than the exhaustive search of than 2^{96} operations. So, it can be concluded that SEA withstand the resistivity against XSL attack.

5.6 Conclusion

In this chapter, the equations systems for SEA S-box have been formulated. Then using individual linear and nonlinear component's equations, algebraic representations of 33 rounds of $SEA_{48,8}$ has been presented in terms of 10, 560 equations in 3216 variables with 34,320 terms. Moreover, algebraic representation of $SEA_{96,8}$ terms of 6432 variables, 21,120 equations and 68, 568 monomials has also been presented. Since the best known attack against block cipher is on 6 round DES [4] through solving systems involving 2900 variables, 3030 equations in 4331 monomials, therefore, both variant of SEA are adequately resistant algebraic attacks in feasible time. Moreover, if XSL attack works as claimed [44], SEA is adequately secure against it.

Chapter 6

AIDA/CUBE Attack and Algebraic Attack

6.1 Introduction

Among various academic objectives targeted for this research work, one objective is to undertake feasibility study, in light of published literature, about the applicability of cube attack in comparison of algebraic attack on feistel structure based light weight block ciphers. Same has been described in this chapter. Chapter 6 has been divided into 3 sections. Section 6.2 contains basic concept of cube attack with reference to block ciphers. Section 6.3 presents feasibility about applicability of cube attack along with algebraic attack on light weight block ciphers in view of published literature. Section 6.4 concludes the chapter.

6.2 Cube Attack

Cube Attack is new type of cryptanalytic technique introduced in [57] by Adi Shamir and Dinur that treats cipher as a black box. Ciphers whose output bit (cipher text bits) can be represented in terms of key and input bits (plaintext /IV) with very low degree polynomial over GF(2) are extremely vulnerable against this cryptanalytic attack. This attack has two phases; namely, preprocessing and an online phase. Prior discussing the core concept behind cube attack, following definitions have been taken from [57, 58, 59].

6.2.1 Definition 6.1

Suppose some polynomial $p(x_1, \dots, x_n)$ and a set of indices $I \subseteq (1, \dots, n)$ to any variables of p. Consider t_I be a subterm of p which is the product of the variables indexed by I. Thus, factorizing p by t_I gives following:

$$p(x_1, \dots, x_n) \equiv t_I \cdot p_{s(I)} + q(x_1, \dots, x_n) \quad (6.1)$$

In equation 6.1, $p_{s(I)}$ is called superpoly of I in p, and q is called the linear combination of all terms which do not contain t_I.

6.2.2 Definition 6.2

Term t_I is called maxterm with an associated superpoly $p_{s(I)}$ such that $\deg(p_{s(I)}) = 1$, therefore, the superpoly of I in p is a non-constant linear polynomial.

In light of abovementioned definitions, theoretical concept behind cube attack is explained with the help of prototype cipher example in this section.

6.2.3 The Preprocessing Phase

In order to describe basic concept of the cube attack, a prototype cipher is considered with a master polynomial p of degree $d = 3$, having three private variables or key bits (x_1, x_2, x_3) and three public variables or plaintext /IV bits (v_1, v_2, v_3) in the ANF form.

$$p(v_1, v_2, v_3, x_1, x_2, x_3) = v_1v_2x_1 + v_1v_3x_1 + v_2v_3x_1 + v_1v_2x_3 + v_1v_3x_2 + v_2v_3x_2 + v_1v_3x_3 + v_1x_1x_3 + v_3x_2x_3 + x_1x_2x_3 + v_1v_2 + v_1x_3 + v_3x_1 + x_1x_2 + x_2x_3 + x_2 + v_1 + v_3 + 1 \quad (6.2)$$

Simply, re-arranging the variables in Equation 6.2, following equation can be obtained:

$$p(v_1, v_2, v_3, x_1, x_2, x_3) = v_1v_2(x_1 + x_3 + 1) + v_1v_3(x_1 + x_2 + x_3) + v_2v_3(x_1 + x_2) + v_1x_1x_3 + v_3x_2x_3 + x_1x_2x_3 + v_1x_3 + v_3x_1 + x_1x_2 + x_2x_3 + x_2 + v_1 + v_3 + 1 \quad (6.3)$$

Equation 6.1 can be re-interpreted as shown in Table 6.1.

Table 6.1 Master Polynomial p Expressed as Cube Equation

I	t_I	$p_{s(I)}$	q
(1,2)	v_1v_2	$x_1 + x_3 + 1$	$v_1x_1x_3 + v_3x_2x_3 + x_1x_2x_3 + v_1x_3 + v_3x_1 + x_1x_2 + x_2x_3 + x_2 + v_1 + v_3 + 1$
(1,3)	v_1v_3	$x_1 + x_2 + x_3$	
(2,3)	v_2v_3	$x_1 + x_2$	

If the master polynomial$p$is evaluated over all possible values of $v_1$and v_2 i.e. cube index (1, 2), following derived polynomials are obtained:

$$p(0,0,v_3,x_1,x_2,x_3) = v_3x_2x_3 + x_1x_2x_3 + v_3x_1 + x_1x_2 + x_2x_3 + x_2 + v_3 + 1 \quad (6.4)$$

$$p(0,1,v_3,x_1,x_2,x_3) = v_3x_2 + v_3x_2x_3 + x_1x_2x_3 + x_1x_2 + x_2x_3 + x_2 + v_3 + 1 \quad (6.5)$$

$$p(1,0,v_3,x_1,x_2,x_3) = v_3x_2 + v_3x_3 + x_1x_3 + v_3x_2x_3 + x_1x_2x_3 + x_3 + x_1x_2 + x_2x_3 + x_2 + v_3 \quad (6.6)$$

$$p(1,1,v_3,x_1,x_2,x_3) = v_3 + x_1 + v_3x_3 + x_1x_3 + v_3x_2x_3 + x_1x_2x_3 + v_3x_1 + x_1x_2 + x_2x_3 + x_2 + 1 \quad (6.7)$$

Summing up above derived polynomials, a linear relation $x_1 + x_3 + 1$; in terms of the secret variables is obtained.

Similarly, all possible values of $v_1$and v_3 (cube index 1, 3), generates four derived polynomials [57, 58, 59]. Summation of these derived polynomials results in a linear relationship of secret variables i.e. $x_1 + x_2 + x_3$. Furthermore, all possible values of $v_2$and v_3 (cube index 2, 3), also gives four derived polynomials, and addition of these derived polynomials gives $x_1 + x_2$ as an output.

6.3 The Online Phase

Linear expression obtained from the pre-processing phase can be converted into set of linearly independent equations. Therefore, resultant systems of equations can be solved to determine the unknown key (x_1, x_2, x_3). In order to find out the right hand side of equations obtained from pre-processing phase, cipher is run in online phase with the same process of formulating the equations i.e. same set of public variables and either 0/1 of the unknown key. If master polynomial is tweaked with cube index(1,2) while keeping $v_3 = 0$, the four 0/1 values of the derived polynomials are obtained. Addition of these four values gives the right hand side of the respective equations. Similarly, the online process is repeated for the cube indexes (1, 3) and (2, 3). Consequently, following system of linear equations for unknown key bits is obtained:

$$x_1 + x_3 + 1 = 1 \tag{6.8}$$

$$x_1 + x_2 + x_3 = 0 \tag{6.9}$$

$$x_1 + x_2 = 1 \tag{6.10}$$

Now, it is very easy to solve above systems of equations to determine the unknown key bits. So in this case, recovered key bits are 1, 0 and 1.

6.3 Combining Cube and Algebraic Attacks on Light Weight Block Ciphers

Both algebraic and cube attacks are based on entirely different concept. So applicability through simple amalgamation on any block cipher seems infeasible. Nonetheless, if both are independently applied on any block cipher, resultant complexity though is the addition of individual complexity, but can be much lower than the individual attack complexity as shown in [59] and explained in this section.

The major drawback with the algebraic attack is that they are slower on a large number of rounds. While in case of cube attack, maxterm becomes of very large degree with the increase in number of rounds. Thus finding a linear super polynomial becomes very difficult. Nonetheless, if fewer linear super polynomials are found, so in that case corresponding key bits can easily be recovered. This can help in reducing the complexity of standard algebraic attack. Same concept has been applied against cryptanalysis of KATAN family of light weight block cipher in [58]. It was shown that using 3-bit condition on key bits for 71 round of KATAN32, complexity of cube attack alone is $2^{29.58}$. While the complexity of standard algebraic attack is $2^{66.60}$. However, if cube attacks find the 3 bit key, this reduces the complexity of algebraic attacks to $2^{63.60}$. Because in that case the number of key bits required to be guessed reduce from 35 to 32. However, this needs to tested against other ciphers as well, since no results about applicability of combining algebraic with cube attack on any other cipher has been published in the literature.

6.4 Conclusion

In this chapter, theoretical concept of cube attack has been discussed. Later on in light of published literature, it has been described that though both attacks have different modus operandi, but still combinational approach can help in reducing the overall complexity. Although results of cube attack are not very encouraging against larger number of rounds for block cipher, still they can be helpful in recovering few key bits. This can reduce the complexity of standard algebraic attack since it entails guessing less number of bits; that has been determined through cube attack. Nonetheless, resultant complexity shall be the summation of individual complexities of both the attacks.

Chapter 7

Conclusion and Future Work

7.1 Introduction

In this chapter, thesis has been concluded. Chapter has been organized into 4 sections. Section 7.2 presents an overview of research. Section 7.3 contains some achievements that have accrued as an outcome of this work. Section 7.4 indicates some directions and recommendations for future work. Section 7.5 concludes the chapter.

7.2 Overview of the Research

Since the advent of light weight block ciphers, many cryptanalytic techniques have been taken into account with an aim to analyze the cipher resistivity and improve their design. Since feistel structure based light weight block ciphers don't have any particular algebraic structure, therefore, considerable research regarding their algebraic representation has not been undertaken in the past. This research has been focused on algebraic representation and analysis of feistel structure based light weight schemes, that don't have any particular algebraic structure. In the beginning, a comprehensive literature review regarding light weight block cipher and algebraic cryptanalysis has been presented with special emphasis on methodological application of algebraic cryptanalytic technique against block ciphers. Later on overview of various tools and techniques used in algebraic cryptanalysis with respect to block cipher have been elaborated. Then a step by step methodology for formulating system of equations for any S-box has been described and a software tool has been developed that can

formulate systems of equations for any lower order S-box. After formulating the system of equation describing individual components of cipher, algebraic representation and analysis of LBlock and two version of SEA has been given.

7.3 Achievements

During this research work, a systematic methodology for formulating algebraic representation of any S-box has been described. Then based on it a software tool has been developed in C-sharp and Maple, that researchers can use for algebraic representation of any lower order S-box for not only cryptanalysis purposes of other block ciphers but also for S-box design . Coding has been done in a modular way to facilitate its extension for any higher order s-box very easily. We have shown the developed tools give simpler and sparse system then the previously published results. Using same tool, algebraic representation of complete LBlock in terms of 2628 variables, 8928 equations and 43,908 monomials, 33 rounds of $SEA_{48,8}$ in terms of 3216 variables, 10,560 equations in 34,320 monomials and $SEA_{96,8}$ in terms of 6432 variables, 21,120 equations and 68, 568 monomials have been given. It has been shown that all these ciphers are adequately secure as for practical algebraic attacks are concerned. Resistivity of LBlock and SEA against XSL attack has also been calculated with theoretical work factor more than the exhaustive search.

7.4 Future Work

Using the developed system of equations for LBlock and SEA, practical attack against reduced round can be a good contribution. Based on statistical analysis of individual S-box equations for LBlock and SEA, we have shown that if one input of s-box can be guessed somehow, the resultant system of equations

can become very simple. This can be further exploited. Moreover, using developed tool for algebraic representation of S-boxes, researcher can analyze any other block cipher resistivity, as well.

7.5 Conclusion

In this chapter, overview of research methodology has been described. Then few achievements acquired through this work have been listed. The chapter has been concluded with few directions for future research work.

Appendix A

Output Combination for 3x3 Sbox with XOR Sum Zero

Two Combinations for which XOR is zero:

Three Combinations for which XOR is zero:

---

x2y0 + x1y1 + x1y0

x2y1 + x0y0 + y1

x1y2 + x0y1 + x0

x2y2 + y1 + x0

Four Combinations for which XOR is zero:

---

x1y2 + x1y1 + x0y2 + x0y0

x2y2 + x1y2 + x0y1 + y1

x2y2 + x1y0 + y2 + y0

x2y1 + x2y0 + x1y2 + x2

x1y0 + x0y2 + y1 + x2

x2y2 + x2y1 + x0y0 + x0

Five Combinations for which XOR is zero:

---

x2y2 + x2y1 + x1y1 + x0y2 + x0y1

x2y0 + x1y2 + x1y0 + x0y2 + x0y0

x2y2 + x2y1 + x1y2 + x0y1 + x0y0

x2y1 + x1y2 + x1y1 + x0y2 + y1

x2y2 + x2y0 + x1y1 + y2 + y0

x2y1 + x1y2 + x1y1 + x1y0 + x2

x2y1 + x1y0 + x0y2 + x0y0 + x2

x2y2 + x2y0 + x0y1 + x0y0 + x2

x2y0 + x1y1 + x0y2 + y1 + x2

x2y0 + x1y2 + x0y0 + y1 + x2

x1y1 + x0y2 + x0y1 + x0y0 + x0

x1y0 + y2 + y1 + y0 + x0

x2y2 + x1y0 + x0y2 + x2 + x0

x2y1 + x2y0 + x0y1 + x2 + x0

x0y2 + y2 + y0 + x2 + x0

x2y2 + x2y1 + y2 + x1 + 1

x2y1 + x1y0 + y0 + x1 + 1

x0y0 + y2 + x1 + x0 + 1

Six Combinations for which XOR is zero:

---

x2y2 + x2y1 + x2y0 + x1y0 + x0y2 + x0y1

x2y1 + x2y0 + x1y2 + x1y0 + x0y2 + y1

x2y1 + x2y0 + x1y1 + x1y0 + x0y0 + y1

x2y2 + x1y1 + x0y2 + x0y1 + x0y0 + y1

x2y1 + x2y0 + x0y2 + x0y1 + y2 + y0

x1y2 + x1y0 + x0y1 + y2 + y1 + y0

x2y2 + x1y2 + x1y0 + x0y2 + x0y1 + x2

x2y1 + x2y0 + x1y1 + x0y2 + x0y0 + x2

x2y2 + x1y1 + x1y0 + x0y1 + x0y0 + x2

x2y2 + x2y1 + x2y0 + x0y1 + y1 + x2

x1y2 + x1y1 + x1y0 + x0y0 + y1 + x2

x1y2 + x0y2 + x0y1 + y2 + y0 + x2

x1y1 + x0y1 + x0y0 + y2 + y0 + x2

x2y2 + x0y2 + y2 + y1 + y0 + x2

x2y2 + x2y1 + x1y2 + x1y1 + x0y2 + x0

x2y0 + x1y2 + x1y1 + x1y0 + x0y1 + x0

x2y0 + x1y0 + x0y2 + x0y1 + x0y0 + x0

x2y2 + x2y0 + x1y1 + x1y0 + y1 + x0

x2y1 + x1y1 + x0y2 + x0y1 + y1 + x0

x2y1 + x1y2 + x0y1 + x0y0 + y1 + x0

x2y1 + x1y0 + x0y0 + y2 + y0 + x0

x2y0 + x1y1 + y2 + y1 + y0 + x0

x2y2 + x2y0 + x1y1 + x0y2 + x2 + x0

x2y1 + x1y1 + x1y0 + x0y1 + x2 + x0

x2y2 + x2y0 + x1y2 + x0y0 + x2 + x0

x2y0 + x0y1 + x0y0 + y1 + x2 + x0

x1y1 + x0y2 + x0y1 + y2 + x1 + 1

x1y2 + x0y1 + x0y0 + y2 + x1 + 1

x2y2 + x0y0 + y2 + y1 + x1 + 1

x2y1 + x2y0 + x1y1 + y0 + x1 + 1

x1y0 + x0y0 + y1 + y0 + x1 + 1

x1y2 + x1y1 + y0 + x2 + x1 + 1

x0y2 + x0y0 + y0 + x2 + x1 + 1

x2y1 + y2 + y1 + x1 + x0 + 1

Seven Combinations for which XOR is zero:

---

x2y2 + x2y0 + x1y2 + x1y1 + x1y0 + x0y1 + y1

x2y2 + x2y0 + x1y0 + x0y2 + x0y1 + x0y0 + y1

x2y1 + x1y1 + x1y0 + x0y2 + x0y1 + y2 + y0

x2y2 + x2y0 + x1y2 + x0y2 + x0y0 + y2 + y0

x2y1 + x1y2 + x1y0 + x0y1 + x0y0 + y2 + y0

x2y0 + x1y2 + x1y1 + x0y1 + y2 + y1 + y0

x2y2 + x2y1 + x1y0 + x0y0 + y2 + y1 + y0

x2y0 + x0y2 + x0y1 + x0y0 + y2 + y1 + y0

x2y2 + x2y0 + x1y2 + x1y1 + x0y2 + x0y1 + x2

x2y2 + x2y1 + x1y1 + x1y0 + x0y1 + y1 + x2

x2y2 + x2y1 + x1y2 + x1y1 + y2 + y0 + x2

x2y2 + x2y1 + x0y2 + x0y0 + y2 + y0 + x2

x2y0 + x1y0 + x0y1 + x0y0 + y2 + y0 + x2

x2y1 + x1y1 + x0y1 + y2 + y1 + y0 + x2

x2y2 + x2y1 + x2y0 + x1y2 + x1y0 + x0y2 + x0

x2y2 + x2y1 + x2y0 + x1y1 + x1y0 + x0y0 + x0

x2y1 + x2y0 + x1y0 + x0y2 + x0y1 + y1 + x0

x2y2 + x1y2 + x1y1 + x0y2 + x0y0 + y1 + x0

x2y1 + x2y0 + x1y2 + x0y2 + y2 + y0 + x0

x2y2 + x1y2 + x1y0 + x0y1 + y2 + y0 + x0

x2y1 + x2y0 + x1y1 + x0y0 + y2 + y0 + x0

x2y2 + x1y2 + x1y1 + x1y0 + x0y0 + x2 + x0

x2y2 + x2y1 + x2y0 + x1y2 + y1 + x2 + x0

x1y2 + x1y0 + x0y2 + x0y1 + y1 + x2 + x0

x1y1 + x1y0 + x0y1 + x0y0 + y1 + x2 + x0

x1y2 + x1y1 + x0y0 + y2 + y0 + x2 + x0

x2y0 + x1y0 + x0y2 + x0y1 + y2 + x1 + 1

x2y1 + x1y2 + x0y1 + y2 + y1 + x1 + 1

x2y2 + x2y0 + x0y2 + x0y1 + y0 + x1 + 1

x2y0 + x1y2 + x0y2 + y1 + y0 + x1 + 1

x2y0 + x1y1 + x0y0 + y1 + y0 + x1 + 1

x2y2 + x2y0 + x1y2 + y2 + x2 + x1 + 1

x2y0 + x0y1 + y2 + y1 + x2 + x1 + 1

x2y0 + x1y2 + x1y0 + y0 + x2 + x1 + 1

x2y1 + x0y2 + y1 + y0 + x2 + x1 + 1

x1y2 + x1y1 + x0y2 + y2 + x1 + x0 + 1

x2y2 + x1y0 + x0y0 + y0 + x1 + x0 + 1

x1y1 + x0y1 + y0 + x2 + x1 + x0 + 1

Appendix B

LBlock S-BOX Equations

Equations for LBlock S1

y2y3 + x0x3 + x2y3 + x1y1

y1y3 + x3y1 + y1

y1y2 + x3y2 + x2y2 + x1y2 + x0y2

y0y3 + x0x1 + x2y2 + x0y1

y0y2 + x3y0 + x2y0

y0y1 + x3y1 + x2y1 + x0y1

x2x3 + x3y2 + x3

x1x3 + x0x3 + x3y1 + x3

x1x2 + x0x2 + x2y1 + x2

x0x3 + x0x1 + x2y2 + x1y1 + x0y1 + y3

x0x2 + x0x1 + x2y3 + x2y2 + x0y1 + x0y0

x0x1 + x3y0 + x2y2 + x0y1 + y0

x3y3 + x3y0 + x1y1 + x0y3 + y3 + y0 + x0

x3y2 + x3y0 + x2y3 + x2y1 + x2y0 + x0y0 + y0 + x3

x3y1 + x0y1 + y3 + y1 + x0

x3y0 + y2 + y0 + x2 + 1

x2y3 + x2y2 + x2y1 + x2y0 + x1y2 + x1y1 + x0y3 + x0y2 + x0y1 + x0y0 + y3 +

y2 + y1 + y0 + x2 + x0 + 1

x2y2 + x2y1 + x1y0 + x0y0 + y3 + y2 + y1 + y0 + x2 + x0 + 1

x2y1 + y0 + x3 + x0

x2y0 + x1y3 + x1y2 + x1y1 + x0y3 + y1 + x3

x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y2 + y1 + y0 + x1 + 1

Equations for LBlock S2

y2y3 + x3y3 + y3

y1y3 + x3y1 + x2y1 + x0y1

y1y2 + x3y1 + y1

y0y3 + x3y3 + x2y3

y0y2 + x0x3 + x2y2 + x1y1

y0y1 + x3y0 + x2y0 + x1y0 + x0y0

x2x3 + x3y0 + x3

x1x3 + x0x3 + x3y1 + x3

x1x2 + x3y0 + x2y3 + x3 + x2

x0x3 + x3y3 + x1y1 + y3 + y2

x0x2 + x3y3 + x2y2 + x0y3 + y3

x0x1 + x3y3 + x2y0 + x0y1 + y3

x3y3 + x3y0 + x2y3 + x2y2 + x2y1 + x0y3 + y3 + x3

x3y2 + x3y0 + x2y3 + x2y2 + x2y1 + x1y1 + x0y3 + x0y2 + y2 + x3 + x0

x3y1 + x0y1 + y2 + y1 + x0

x3y0 + x2y3 + x2y2 + x2y1 + x0y3 + y0 + x3 + x2 + 1

x2y3 + x2y2 + x2y1 + x2y0 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y1 + y0 + x2 + x0 + 1

x2y2 + x2y1 + x2y0 + x1y2 + x0y3 + x0y1 + x0y0 + y3 + y2 + y0 + x3 + x2 + x0 + 1

x2y1 + y3 + x3 + x0

x2y0 + x1y3 + x0y3 + y2 + y1 + y0 + x3 + x2 + 1

x1y3 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y1 + y0 + x1 + 1

Equations for LBlock S3

y2y3 + x3y2 + x2y2

y1y3 + x0x3 + x2y1 + x1y0

y1y2 + x3y2 + y2

y0y3 + x2y3 + x1y3 + x0y3 + y3

y0y2 + y0y1 + x1x3 + x3y2 + x1y0

y0y1 + x1x3 + x3y2 + x2y3 + x1y2 + x1y0 + x0y2 + y2

x2x3 + x0x2 + x2y2

x1x3 + x0x3 + x3y0 + x3

x1x2 + x0x3 + x3y2 + x3y0 + x2y2 + x1y0 + x0y0 + x3

x0x3 + x0x2 + x2y1 + x1y0 + x0y2 + x0

x0x2 + x3y3 + x2y2 + x3

x0x1 + x2y3 + x0y0

x3y3 + x3y1 + x2y2 + x2y1 + x1y0 + x0y2 + x0y1 + x3

x3y2 + y3 + y2 + x2 + 1

x3y1 + x2y3 + x1y3 + x0y3 + x0y0 + y2 + x3 + x2 + 1

x3y0 + x2y3 + x2y0 + x1y2 + x0y2 + x0y0 + y2 + y0

x2y3 + x2y2 + x2y1 + x2y0 + x1y2 + y3 + y2 + y0 + x1 + x0

x2y2 + x2y1 + x1y2 + x1y0 + x0y2 + x0y0 + y1 + x3 + x0

x2y1 + x2y0 + x1y1 + x1y0 + x0y3 + y3 + y2 + y1 + y0 + x3 + x1 + x0

x2y0 + y2 + y0 + x3 + x0

x1y1 + x0y1 + x0y0 + y0 + x1 + 1

Equations for LBlock S4

y2y3 + x3y2 + x2y2

y1y3 + x0x3 + x2y1 + x1y0

y1y2 + x3y2 + y2

y0y3 + x2y3 + x1y3 + x0y3 + y3

y0y2 + y0y1 + x1x3 + x3y2 + x1y0

y0y1 + x0y0 + y1 + x0

x2x3 + x0x2 + x2y2

x1x3 + x0x3 + x3y0 + x3

x1x2 + x0x3 + x3y2 + x3y0 + x2y2 + x1y0 + x0y0 + x3

x0x3 + x3y2 + x1y0 + y2 + y1

x0x2 + x3y3 + x2y2 + x3

x0x1 + x2y3 + x0y0

x3y3 + x3y2 + x2y2 + x2y1 + x0y2 + y2 + y1 + x3 + x0

x3y2 + x3y1 + x1y0 + x0y1 + y2 + y1 + x0

x3y1 + x1y0 + x0y1 + y3 + y1 + x2 + x0 + 1

x3y0 + x2y3 + x1y2 + x0y2 + x0y0 + x3 + x0

x2y3 + x1y3 + x1y0 + x0y3 + x0y1 + x0y0 + y3 + y2 + y1 + x3 + x0

x2y2 + x2y1 + x1y3 + x1y2 + x1y0 + x0y3 + x0y1 + x0y0 + y2 + y1 + x1 + x0

x2y1 + x1y3 + x1y1 + x1y0 + x0y2 + x0y1 + y3 + y2 + y1 + x3

x2y0 + y2 + y0 + x3 + x0

x1y1 + x0y1 + x0y0 + y0 + x1 + 1

Equations for LBlock S5

y2y3 + x3y2 + y2

y1y3 + x0x3 + x2y3 + x1y0

y1y2 + x3y2 + x2y2

y0y3 + x3y0 + y0

y0y2 + x3y0 + x2y0 + x0y0

y0y1 + x3y1 + x2y1 + x1y1 + x0y1

x2x3 + x0x2 + x2y2 + x2y0

x1x3 + x0x3 + x3y0 + x3

x1x2 + x0x2 + x2y0 + x2

x0x3 + x3y2 + x1y0 + y3 + y2

x0x2 + x3y2 + x2y3 + x0y2 + y2

x0x1 + x3y2 + x2y1 + x0y0 + y2

x3y3 + x3y2 + x1y0 + x0y3 + y3 + y2 + x0

x3y2 + y2 + y1 + x2 + 1

x3y1 + x2y3 + x2y2 + x2y0 + x0y2 + y1 + x3 + x2 + 1

x3y0 + x0y0 + y3 + y0 + x0

x2y3 + x2y2 + x2y1 + x2y0 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 +

y2 + y1 + y0 + x2 + x0 + 1

x2y2 + x1y3 + x1y1 + x1y0 + x0y3 + y0 + x3

x2y1 + x2y0 + x1y2 + x0y2 + y3 + y2 + y1 + y0 + x2 + x0 + 1

x2y0 + y2 + x3 + x0

x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y2 + y1 + y0 + x1 + 1

Equations For LBlock S6

y2y3 + x0x3 + x2y3 + x1y1

y1y3 + x3y1 + y1

y1y2 + x2y2 + x1y2 + x0y2 + y2

y0y3 + x3y0 + y0

y0y2 + x3y0 + x2y0

y0y1 + x3y1 + x2y1 + x0y1 + y1

x2x3 + x3y2 + x3

x1x3 + x0x3 + x3y1 + x3

x1x2 + x0x2 + x3y2 + x2y1 + x3

x0x3 + x3y0 + x1y1 + y3 + y0

x0x2 + x3y2 + x2y0 + x3

x0x1 + x2y2 + x0y1

x3y3 + x3y0 + x1y1 + x0y3 + y3 + y0 + x0

x3y2 + x3y0 + x2y3 + x2y0 + x0y0 + y3 + y0 + x3 + x0

x3y1 + x0y1 + y3 + y1 + x0

x3y0 + y2 + y0 + x2 + 1

x2y3 + x2y1 + x2y0 + x0y0 + y3 + y2 + y0 + x3 + x1 + x0

x2y2 + x2y1 + x1y0 + x0y0 + y3 + y0 + x0

x2y1 + y1 + y0 + x3 + x0

x2y0 + x1y3 + x1y0 + x0y2 + x0y1 + x0y0 + y2 + x3 + x1 + x0

x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y2 + y1 + y0 + x0

Equations for LBlock S7

y2y3 + y0y3 + x2y3

y1y3 + x3y1 + y1

y1y2 + x2y2 + x1y2 + x0y2

y0y3 + x3y0 + y0

y0y2 + x2x3 + x1x2 + x3y0 + y0

y0y1 + x0x3 + x3y0 + x1y1

x2x3 + x1x2 + x2y0

x1x3 + x3y3 + x0y3 + x0y1

x1x2 + x3y2 + x2y0

x0x3 + x3y1 + x3y0 + x2y1 + x1y1 + x0y1 + y1

x0x2 + x3y2 + x2y1 + x2y0 + x2

x0x1 + x3y2 + x2y1 + x2y0 + x1y3 + x0y2 + x2 + x1

x3y3 + x3y0 + x2y1 + x1y1 + x0y3 + y1 + x3

x3y2 + x3y0 + x2y1 + x2y0 + x1y3 + x1y2 + x1y1 + x0y3 + y1 + x3

x3y1 + x3y0 + x2y3 + x2y2 + x2y1 + x1y3 + x1y2 + x1y1 + x1y0 + x0y3 + x0y1 + x3 + x2

x3y0 + y2 + y0 + x2

x2y3 + x1y3 + x1y2 + x1y1 + x0y3 + x0y0 + y1 + x3

x2y2 + x2y1 + x1y2 + x1y1 + x0y3 + x0y2 + x0y1

x2y1 + x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y1 + x3

x2y0 + x1y3 + x1y0 + x0y2 + x0y1 + x0y0 + y3 + y1 + y0 + x3 + x1 + x0

x1y2 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y1 + y0 + x0 + 1

Equations for LBlock S8

y2y3 + x0x3 + x2y2 + x1y1

y1y3 + x3y3 + x2y3 + x1y3 + x0y3

y1y2 + x0y1 + y2 + x0

y0y3 + x3y0 + x2y0

y0y2 + x0x1 + x2y3 + x0y1

y0y1 + x3y1 + x2y1 + x0y1

x2x3 + x0x2 + x2y1 + x2y0

x1x3 + x0x3 + x3y1 + x3

x1x2 + x0x2 + x2y1 + x2

x0x3 + x0x1 + x2y3 + x1y1 + x0y1 + y2

x0x2 + x0x1 + x2y3 + x2y2 + x0y1 + x0y0

x0x1 + x3y0 + x2y3 + x0y1 + y0

x3y3 + x3y0 + x2y2 + x2y1 + x2y0 + x0y0 + y0 + x3

x3y2 + x3y0 + x1y1 + x0y2 + y2 + y0 + x0

x3y1 + x0y1 + y2 + y1 + x0

x3y0 + y3 + y0 + x2 + 1

x2y3 + x2y1 + x1y0 + x0y0 + y3 + y2 + y1 + y0 + x2 + x0 + 1

x2y2 + x2y0 + x1y3 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1

x2y1 + y0 + x3 + x0

x2y0 + x1y3 + x1y2 + x1y1 + x0y2 + y1 + x3

x1y3 + x1y1 + x1y0 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y1 + y0 + x1 + 1

Equations for LBlock S9

y2y3 + x0x3 + x3y3 + x1y2

y1y3 + y0y3 + x2y3

y1y2 + x2y1 + x1y1 + x0y1

y0y3 + y0y1 + x2y0

y0y2 + x3y2 + y2

y0y1 + x3y3 + x2y0 + y3

x2x3 + x1x2 + x2y3

x1x3 + x3y0 + x0y2 + x0y0

x1x2 + x3y1 + x2y3

x0x3 + x3y3 + x3y2 + x2y2 + x1y2 + x0y2 + y2

x0x2 + x3y1 + x2y3 + x2y2 + x2

x0x1 + x3y1 + x2y3 + x2y2 + x1y0 + x0y1 + x2 + x1

x3y3 + y3 + y1 + x2

x3y2 + x3y1 + x2y3 + x2y1 + x2y0 + x1y3 + x0y2 + y2 + x2

x3y1 + x3y0 + x2y3 + x1y1 + x1y0

x3y0 + x2y2 + x1y2 + x0y0 + y3 + y2 + y1 + x3 + x2

x2y3 + x2y2 + x1y2 + x1y1 + x1y0 + x0y0 + y3 + y1 + x1 + x0

x2y2 + x2y1 + x1y2 + x1y1 + x0y2 + x0y1 + x0y0

x2y1 + x2y0 + x1y3 + x1y2 + x1y1 + x1y0 + x0y0 + y1 + y0

x2y0 + x1y2 + x1y1 + x1y0 + x0y3 + x0y0 + y2 + x3

x1y3 + x1y2 + x1y1 + x0y3 + x0y2 + x0y1 + x0y0 + y3 + y2 + y1 + y0 + x0 + 1

Appendix C

Statistical Analysis of Individual S-box equations –LBlock

SBOX 0

Monomial	Count	Percent
x1y0	10	7.41%
x2y0	9	6.67%
x0y3	9	6.67%
x2y1	8	5.93%
x1y2	8	5.93%
y0	7	5.19%
x0y0	6	4.44%
x3y1	6	4.44%
x0y2	5	3.70%
x1y3	5	3.70%
y2	5	3.70%
y1	5	3.70%
x3	5	3.70%
x3y2	4	2.96%
x2y3	3	2.22%
x2	3	2.22%
x3y0	3	2.22%
x0y1	3	2.22%
x1y1	3	2.22%
x2y2	3	2.22%
x0x3	2	1.48%
y3	2	1.48%
x1	2	1.48%
x1x2	2	1.48%
y1y2	2	1.48%
x0	2	1.48%
x3y3	2	1.48%
y1y3	2	1.48%
y0y1	1	0.74%
y0y2	1	0.74%
y0y3	1	0.74%
x2x3	1	0.74%
x1x3	1	0.74%
x0x2	1	0.74%
x0x1	1	0.74%
y2y3	1	0.74%
1	1	0.74%

SBOX1

Monomial	Count	Percent
x0y1	8	6.30%
y0	8	6.30%
x2y2	7	5.51%
y1	6	4.72%
x1y1	6	4.72%
x2y1	6	4.72%
x3y0	5	3.94%
x0y0	5	3.94%
y3	5	3.94%
x3	5	3.94%
x0	5	3.94%
y2	4	3.15%
x2y3	4	3.15%
1	4	3.15%
x1y2	4	3.15%
x0y3	4	3.15%
x2	4	3.15%
x3y1	4	3.15%
x0x1	4	3.15%
x2y0	4	3.15%
x3y2	3	2.36%
x0y2	3	2.36%
x0x3	3	2.36%
x1y0	2	1.57%
x0x2	2	1.57%
x2x3	1	0.79%
x3y3	1	0.79%
x1x3	1	0.79%
x1y3	1	0.79%
y0y2	1	0.79%
y0y3	1	0.79%
x1x2	1	0.79%
y1y2	1	0.79%
y1y3	1	0.79%
x1	1	0.79%
y2y3	1	0.79%
y0y1	1	0.79%

SBOX2

Monomial	Count	Percent
y3	9	6.52%
x3	9	6.52%
x0y3	8	5.80%
x2y2	7	5.07%
x2y1	7	5.07%
y2	6	4.35%
x3y3	6	4.35%
x0y1	6	4.35%
x3y0	6	4.35%
x2y3	6	4.35%
y0	5	3.62%
1	5	3.62%
y1	5	3.62%
x1y1	5	3.62%
x0	5	3.62%
x2	5	3.62%
x2y0	5	3.62%
x0y0	4	2.90%
x3y1	4	2.90%
x1y0	3	2.17%
x0x3	3	2.17%
x0y2	3	2.17%
x1y3	2	1.45%
y0y3	1	0.72%
x1	1	0.72%
x3y2	1	0.72%
x2x3	1	0.72%
x1x2	1	0.72%
x1x3	1	0.72%
y0y2	1	0.72%
x1y2	1	0.72%
x0x2	1	0.72%
y1y2	1	0.72%
y1y3	1	0.72%
x0x1	1	0.72%
y2y3	1	0.72%
y0y1	1	0.72%

SBOX3

Monomial	Count	Percent
x3	8	6.25%
y2	8	6.25%
x1y0	8	6.25%
x2y2	7	5.47%
x0y0	6	4.69%
x3y2	6	4.69%
x2y3	6	4.69%
x2y1	6	4.69%
x0y2	5	3.91%
x0	5	3.91%
y0	5	3.91%
x2y0	4	3.13%
x0x3	4	3.13%
y3	4	3.13%
x1y2	4	3.13%
1	3	2.34%
x1x3	3	2.34%
x1	3	2.34%
x0y3	3	2.34%
x3y0	3	2.34%
x0x2	3	2.34%
y1	2	1.56%
x1y1	2	1.56%
x1y3	2	1.56%
x3y1	2	1.56%
x0y1	2	1.56%
x2	2	1.56%
y0y1	2	1.56%
x3y3	2	1.56%
x1x2	1	0.78%
y0y2	1	0.78%
y0y3	1	0.78%
x2x3	1	0.78%
y1y2	1	0.78%
y1y3	1	0.78%
x0x1	1	0.78%
y2y3	1	0.78%

SBOX 4

Monomial	Count	Percent
x1y0	9	7.14%
x3	8	6.35%
x0	8	6.35%
y2	8	6.35%
y1	8	6.35%
x0y0	7	5.56%
x3y2	7	5.56%
x0y1	6	4.76%
x2y2	6	4.76%
x1y3	4	3.17%
x2y1	4	3.17%
x2y3	4	3.17%
x0x3	4	3.17%
y3	4	3.17%
x0y3	3	2.38%
x0y2	3	2.38%
x3y0	3	2.38%
y0	2	1.59%
x0x2	2	1.59%
x1x3	2	1.59%
1	2	1.59%
x1y1	2	1.59%
x1y2	2	1.59%
x1	2	1.59%
x3y1	2	1.59%
y0y1	2	1.59%
x3y3	2	1.59%
x2x3	1	0.79%
x2	1	0.79%
y0y2	1	0.79%
y0y3	1	0.79%
x1x2	1	0.79%
y1y2	1	0.79%
y1y3	1	0.79%
x0x1	1	0.79%
y2y3	1	0.79%
x2y0	1	0.79%

SBOX5

Monomial	Count	Percent
y2	10	7.94%
x2y0	7	5.56%
x3y2	7	5.56%
y0	6	4.76%
x1y0	6	4.76%
x0y0	5	3.97%
y1	5	3.97%
x0y2	5	3.97%
x2y2	5	3.97%
x0	5	3.97%
x2	5	3.97%
1	5	3.97%
y3	5	3.97%
x1y1	4	3.17%
x2y1	4	3.17%
x0y3	4	3.17%
x2y3	4	3.17%
x3	4	3.17%
x3y0	4	3.17%
x0x2	3	2.38%
x0x3	3	2.38%
x0y1	3	2.38%
x1y2	2	1.59%
x3y1	2	1.59%
y0y3	1	0.79%
x1	1	0.79%
x3y3	1	0.79%
x1x2	1	0.79%
y0y1	1	0.79%
y0y2	1	0.79%
x2x3	1	0.79%
x1y3	1	0.79%
y1y2	1	0.79%
y1y3	1	0.79%
x0x1	1	0.79%
y2y3	1	0.79%
x1x3	1	0.79%

SBOX6

Monomial	Count	Percent
y0	9	7.76%
x3	8	6.90%
x0	8	6.90%
y3	6	5.17%
x3y0	6	5.17%
x0y0	5	4.31%
x0y1	5	4.31%
y2	5	4.31%
y1	5	4.31%
x2y1	5	4.31%
x2y0	5	4.31%
x3y1	4	3.45%
x1y1	4	3.45%
x3y2	4	3.45%
x2y2	3	2.59%
x2y3	3	2.59%
x0y2	3	2.59%
x1y0	3	2.59%
x0x3	3	2.59%
x1y2	2	1.72%
x1	2	1.72%
x0x2	2	1.72%
x0y3	2	1.72%
x1x2	1	0.86%
x1x3	1	0.86%
x1y3	1	0.86%
x3y3	1	0.86%
x2	1	0.86%
y0y1	1	0.86%
y0y2	1	0.86%
y0y3	1	0.86%
x2x3	1	0.86%
y1y2	1	0.86%
y1y3	1	0.86%
x0x1	1	0.86%
y2y3	1	0.86%
1	1	0.86%

SBOX7

Monomial	Count	Percent
x1y1	9	6.52%
x2y1	8	5.80%
x3y0	8	5.80%
x0y3	8	5.80%
y1	8	5.80%
x0y1	7	5.07%
x1y2	7	5.07%
x3	6	4.35%
x0y2	6	4.35%
x2y0	6	4.35%
x1y3	5	3.62%
y0	5	3.62%
x3y2	4	2.90%
x2	4	2.90%
x0y0	4	2.90%
x1y0	4	2.90%
y2	3	2.17%
y3	3	2.17%
x1x2	3	2.17%
x3y1	3	2.17%
x2y2	3	2.17%
x2y3	3	2.17%
x0	2	1.45%
x1	2	1.45%
x3y3	2	1.45%
x0x3	2	1.45%
x2x3	2	1.45%
y0y3	2	1.45%
y0y1	1	0.72%
x1x3	1	0.72%
1	1	0.72%
x0x2	1	0.72%
y1y2	1	0.72%
y1y3	1	0.72%
x0x1	1	0.72%
y2y3	1	0.72%
y0y2	1	0.72%

SBOX 8

Monomial	Count	Percent
x0y1	9	7.50%
y0	7	5.83%
x1y1	6	5.00%
x2y3	6	5.00%
x2y1	6	5.00%
y2	5	4.17%
x0	5	4.17%
x2y0	5	4.17%
x3y0	5	4.17%
x1y3	4	3.33%
x0x1	4	3.33%
y1	4	3.33%
x0y2	4	3.33%
x2y2	4	3.33%
x0y0	4	3.33%
x3	4	3.33%
x2	3	2.50%
x0x2	3	2.50%
x0x3	3	2.50%
x1y0	3	2.50%
x0y3	3	2.50%
x3y1	3	2.50%
1	3	2.50%
y3	3	2.50%
x3y3	2	1.67%
x3y2	1	0.83%
x2x3	1	0.83%
x1x3	1	0.83%
y0y1	1	0.83%
y0y2	1	0.83%
x1y2	1	0.83%
x1x2	1	0.83%
y1y2	1	0.83%
y1y3	1	0.83%
x1	1	0.83%
y2y3	1	0.83%
y0y3	1	0.83%

SBOX9

Monomial	Count	Percent
x2y3	8	6.40%
x1y2	8	6.40%
x0y0	7	5.60%
x1y1	7	5.60%
y2	6	4.80%
x2y2	6	4.80%
x2y0	5	4.00%
x0y2	5	4.00%
x3y1	5	4.00%
x2	5	4.00%
y3	5	4.00%
y1	5	4.00%
x1y0	5	4.00%
x2y1	4	3.20%
x0y1	4	3.20%
x3y3	4	3.20%
x3y2	3	2.40%
x1y3	3	2.40%
x3y0	3	2.40%
x1	2	1.60%
x0	2	1.60%
y0y3	2	1.60%
x0x3	2	1.60%
x1x2	2	1.60%
x0y3	2	1.60%
y0	2	1.60%
y0y1	2	1.60%
x3	2	1.60%
x1x3	1	0.80%
y0y2	1	0.80%
x2x3	1	0.80%
x0x2	1	0.80%
y1y2	1	0.80%
y1y3	1	0.80%
x0x1	1	0.80%
y2y3	1	0.80%
1	1	0.80%

Appendix D

User Manual for Software Tool

1. Define monomials based on the I/O degree of the S-Box.
2. Generate input array for defined monomials using Maple Code.
3. Formulate matrix using generated arrays in Maple. Each row of the matrix corresponds to the each defined monomial.
4. Gaussian Elimination Mod 2 of the generated matrix using Maple. Number of non zero rows determines the total equations that shall be generated to completely define the S-box.
5. Convert decimal value for each row of the input array generated in step 1.
6. Insert decimal values in C-sharp code's 'Main Page'.
7. Run the C-sharp code. C-sharp code generates two files, Extended Matrix and XOR Combination Output.
8. Gaussian Eliminate Mod 2 the generated matrix using Maple. This shall output the final matrix that describes the equations.
9. Insert input rows of the matrix in C-sharp code, and run it. It displays the resultant equations that completely describe the pertinent S-Box.
10. Incase of any query please email: nasirjadoon328@gmail.com.

BIBLIOGRAPHY

[1] Bard, G.: Algebraic Cryptanalysis. Springer (2009)

[2] Courtois, N.: General Principles of Algebraic Attacks and New Design Criteria for Components of Symmetric Ciphers. In: AES 4 Conference, Bonn May 10-12 2004, LNCS 3373, pp. 67-83, Springer (2005)

[3] Shanon, C.: Communication Theory of Secrecy Systems, In: Bell Technical Journal 28, pages 656-715, (1949)

[4] Courtois, N., Bard, G.: Algebraic Cryptanalysis of the Data Encryption Standard. In: S.D. Galbraith (ed.) IMA International Conference on Cryptography and Coding Theory, Lecture Notes in Computer Science, vol. 4887, pp. 152–169. Springer-Verlag (2007)

[5] Courtois, N., Bard, G.V., Wagner, D.: Algebraic and Slide Attacks on KeeLoq. In: FSE, Volume 5086 of Lecture Notes in Computer Science, Springer pp. 97-115 (2008)

[6] Bard, G., Courtois, N., N,Jorge Jr., Shepherded, P., Zhang, B.: Algebraic, AIDA/Cube and Side Channel Analysis of KATAN Family of Block Ciphers, In: Progress in Cryptology - INDOCRYPT, Lecture Notes in Computer Science Volume 6498, 2010, pp 176-196 (2010)

[7] Shamir, A., Courtois, N., Patarin, J., Klimov, A.: Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations. In: Advances in Cryptology, Eurocrypt'2000, LNCS 1807, Springer, pp. 392-407 (2000)

[8] Faugere, J., Joux, A.: Algebraic cryptanalysis of Hidden Field Equation (HFE) cryptosystems using Groebner Bases. In: D. Boneh (ed.) Advances in Cryptology—Proc. of CRYPTO, Lecture Notes in Computer Science, vol. 2729, pp. 44–60. Springer-Verlag (2003)

[9] Courtois, N.: Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt, In: Proc. of ICISC, Lecture Notes in Computer Science. Springer-Verlag (2002)

[10] Courtois, N.: The security of cryptographic primitives based on multivariate algebraic problems: MQ, MinRank, IP, HFE. Ph.D. thesis, Paris VI (2001)

[11] Farhadian, A., Aref, M.: Algebraic Cryptanalysis of Reduced AES, In: Proceed. of 3rd Information Security and Cryptology Conference, Ankara, (2008)

[12] Dewu, X., Wei, C.: A Survey on Cryptanalysis of Block Ciphers, In: International Conference on Computer Application and System Modeling (ICCASM), 2010

[13] Cid, C., Murphy, S., Robshaw. M.: Algebraic Aspects of the Advanced Encryption Standard, Springer, (2006)

[14] Wu, W., Zhang, L.: LBlock: A Lightweight Block Cipher, In: Proc. of 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. pp 327-344, (2011)

[15] Xavier, F., Piret, G., Gershenfeld, N., Jacques, Q.: SEA: A Scalable Encryption Algorithm for Small Embedded Applications, In: Springer, CARDIS, volume 3928 of Lecture Notes in Computer Science, pages 222–236, (2006)

[16] Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B., Lee, C., Chang. D., Lee, J., Jeong, K., Kim, H., Kim, J., Chee, S.: South Korea Telecommunications Technology Associations (TTA). 64-bit Block Cipher HIGHT, Standardization Number TTAS.KO-12.0040, 27 December 2006, volume 4249 of Lecture Notes in Computer Science, pages 46–59, Springer, (2006)

[17] Knudsen, L., Leander, G., Poschmann, A., Robshaw, B.: PRINT cipher: A block cipher for IC-printing. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010, LNCS, vol. 6225, pp. 16–32. Springer, Heidelberg (2010)

[18] Lim, H., Korkishk, T.: mCrypton - A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors, In: WISA, volume 3786 of Lecture Notes in Computer Science, pages 243–258. Springer, 2005.

[19] Robshaw, B.: Searching for Compact Algorithms: CGEN, In: Phong Q. Nguyen, editor, VIETCRYPT, volume 4341 of Lecture Notes in Computer Science, pages 37–49, Springer, (2006)

[20] Bogdanov, A., Knudsen, L., Leander, G., Paar, C., Poschmann, A., Robshaw, B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher, In: Pascal Paillier and Ingrid Verbauwhede, editors, CHES, volume 4727 of Lecture Notes in Computer Science, pages 450–466, Springer, (2007)

[21] Leander, G., Paar, C., Poschmann, A., Schramm, K.: New Lightweight DES Variants. In: FSE, volume 4593 of Lecture Notes in Computer Science, pages 196–210, Springer, (2007)

[22] Guo, J., Peyrin, T., Poschmann, A., Robshaw M.: “The LED block cipher,” In; Cryptographic Hardware and Embedded Systems CHES 2011, ser. Lecture

Notes in Computer Science, B. Preneel and T. Takagi, Eds. Springer Berlin Heidelberg, vol. 6917, pp. 326–341 (2011)

[23] Yap, H., Khoo, K., Poschmann, A., Henricksen, M., "EPCBC - a block cipher suitable for electronic product code encryption," In; Cryptology and Network Security, ser. Lecture Notes in Computer Science, D. Lin, G. Tsudik, and X. Wang, Eds. Springer Berlin Heidelberg, vol. 7092, pp. 76–97(2011)

[24] Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai. T.: "Piccolo: An ultra-lightweight blockcipher," In; Cryptographic Hardware and Embedded Systems CHES 2011, ser. Lecture Notes in Computer Science, B. Preneel and T. Takagi, Eds. Springer Berlin Heidelberg, vol. 6917, pp. 342–357, (2011)

[25] Cheng, H., Heys, M., Wang, C.: "Puffin: A novel compact block cipher targeted to embedded digital systems," In; Proceedings of the 2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools, ser. DSD '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 383–390, (2008)

[26] Islam, S., Afzal, M., Rashdi, A.: On the Security of LBlock against the Cube Attack and Side Channel Cube Attack, In: Security Engineering and Intelligence Informatics, Lecture Notes in Computer Science, Volume 8128, 2013, pp 105-121, Springer, (2013)

[27] Soleimany, H., Nyberg, K.: Zero-Correlation Linear Cryptanalysis of Reduced-Round LBlock, In: Designs, Codes and Cryptography, May 2014, Availabe at http://eprint.iacr.org/, Springer (2014)?

[28] Chen, J., Miyaji, A.: Differential Cryptanalysis and Boomerang Cryptanalysis of LBlock, In: Security Engineering and Intelligence Informatics, Lecture Notes in Computer Science Volume 8128, 2013, pp 1-15, Springer (2013)

[29] Wen, L., Wang, M., Zho, J.: Related-Key Impossible Differential Attack on Reduced Round LBlock; In, Journal of Computer Science and Technology January 2014, Volume 29, Issue 1, pp 165-176, Springer (2014)

[30] Emami, S., McDonald.M, Pieprzyk, J., Steinfeld, R., Truncated Differential Analysis of Reduced-Round LBlock, In: Cryptology and Network Security Lecture Notes in Computer Science Volume 8257, 2013, pp 291-308, Springer, (2013)

[31] Mace, F., Standaert, F-X., Quisquater, J.: FPGA Implementation(s) of a Scalable Encryption Algorithm, In; IEEE Transactions on Very Large Scale Integration (VLSI) Systems archive Volume 16 Issue 2, February 2008, Pages 212-216 (2008)

[32] Kumar, P., Ezhumalai, P., Gomathi, S.: Efficient Implementation of a Scalable Encryption Algorithm using FPGA, In; International Journal of Computer Applications (0975 – 8887) Volume 3 – No.10, July 2010 (2010)

[33] Kumar, P., Ezhumalai, P., Gomathi, S., Ramesh, P., Sakthive, P.: Improving the Performance of a Scalable Encryption Algorithm (SEA) using FPGA, In : International Journal of Computer Science and Network Security (IJCSNS), VOL.10 No.2 (2010)

[34] Jegadish, K., Salivahanan, S., Reddy, K.: Implementation of Low Power Scalable Encryption Algorithm, In : International Journal of Computer Applications (0975 – 8887) Volume 11– No.1 (2010)

[35] Courtois, N.: Algebraic Complexity Reduction and Cryptanalysis of GOST, (2011)

[36] Courtois, N.: Security Evaluation of GOST 28147-89 In View Of International Standardization, (2011)

[37] Biruykov, A., Cannιµere, D.: Block Ciphers and Systems of Quadratic Equation, In: Fast Software Encryption, Springer-Verlag, pp. 274, (2003)

[38] De-Canni`ere, C., Dunkelman, O., Kneˇzevi´c, M.: KATAN and KTANTAN — A family of small and efficient hardware-oriented block ciphers. In: CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009)

[39] Gray, M., Johnson, D.: Computers and Intractability: A guide to the Theory of NP-Completeness. W.H.Freeman, (1979)

[40] Soos, M., Nohl, K., Castelluccia, C.: Extending SAT Solvers to Cryptographic Problems, In: Theory and Applications of Satisfiability Testing - SAT 2009, Lecture Notes in Computer Science Volume 5584, 2009, pp 244-257, Springer, (2009)

[41] Courtois, N., Sepehrdad, P., Sušil, P., Vaudenay,V.: ElimLin Algorithem Revisited, In: 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Volume 7549, 2012, pp 306-325 (2012)

[42] Faugere, J.: A new efficient algorithm for computing Groebner basis (F4). Available at http://modular.ucsd.edu/129-05/refs/faugere_f4.pdf, (1999)

[43] Faugere,C.: A New Efficient Algorithm for Computing Groebner Bases without Reduction to Zero (F5), In: Proceedings of ISSAC, ACM Press, pp. 75–83, (2002)

[44] Courtois, N., J, Pieprzyk.: Cryptanalysis of block ciphers with overdefined systems of equations, In: Cryptologia, Available at http://eprint.iacr.org/2002/044, (2002)

[45] Courtois, N.: How fast can be algebraic attacks on block ciphers, Cryptology ePrint Archive, Available at http://eprint.iacr.org/2006/168.pdf (2006)

[46] Albrecht, M.: Algebraic Attacks on the Courtois Toy Cipher, In: Cryptologia, 32:3, 220-276 (2008)

[47] Tang, X., Feng, Y.: A new efficient algorithm for solving systems of multivariate polynomial equations, In: Cryptology ePrint Archive, Report 2005=312. Available at http://eprint. iacr.org/2005/31, (2005)

[48] Ding, J., Gowe, J., Schmidt, S., Zhuang-zi.: A new algorithm for solving multivariate polynomial equations over a finite field, In: Cryptology ePrint Archive, Available at http://eprint.iacr.org/2006/038.pdf, (2006)

[49] Miolane, C.: Block Cipher Analysis, Ph.D. Thesis, Technical University of Denmark (2008)

[50] Garay, J., Miyaji, A., Otsuka, A.: Linear (Hull) and Algebraic Cryptanalysis of the Block Cipher PRESENT, In: CANS, LNCS 5888, pp. 58–75, Springer-Verlag, (2009)

[51] Kreuzer, M.: Algebraic Attacks Galore! Groups-Complexity-Cryptology Volume 1, No. 2, 231–259, (2009)

[52] Christensen, C.: Review of Algebraic Cryptanalysis, In: Cryptologia, 35:1,100-105 (2010)

[53] Babenko, L., Maro, E.: Algebraic Cryptanalysis of GOST Encryption Algorithm, In: Journal of Computer and Communications, 2, 10-17, (2014)

[54] Renauld, M., Standaert, F.X.: Algebraic Side-Channel Attacks, In : 5th International Conference, Inscrypt 2009, Bejing, China 12-15 December 2009, Information Security and Cryptology, Lecture Notes in Computer

Science Volume 6151, 2010, pp 393-410, Springer-Verlag, Also available at http://eprint.iacr.org/, (2010)

[55] Cid, C., Leurent, G.: An Analysis of the XSL Algorithm, In: ASIACRYPT 2005, LNCS 3788, pp. 333–352, (2005)

[56] Izadi, M., Sadeghiyan, B., Sadeghian, S., Khanooki, H.: MIBS: A New Lightweight Block Cipher, CANS 2009, LNCS 5888, pp. 334–348, Springer-Verlag (2009)

[57] Shamir, A., Dinur, I.: Cube Attacks on Tweakable Black Box Polynomials, Available at http://eprint.iacr.org/2008/385.pdf (2008)

[58] Islam, S.: Cube Attacks, MS thesis (2012)

[59] Bard, G., Courtois,N., Nakahara, J., Jorge., Sepehrdad,P., Zhang,B.: Algebraic, AIDA/Cube and side channel analysis of KATAN family of block ciphers, In: INDOCRYPT 2010, Lecture Notes in 86 Computer Science, G. Gong and K. Gupta, Eds. Springer, vol. 6498, pp. 176–196 (2010)