SECURITY ANALYSIS OF BLOCK CIPHERS AND BLOCK CIPHER BASED CONSTRUCTIONS

YAP WUN SHE

DOCTOR OF PHILOSOPHY

MULTIMEDIA UNIVERSITY

AUGUST 2015

SECURITY ANALYSIS OF BLOCK CIPHERS AND BLOCK CIPHER BASED CONSTRUCTIONS

BY

YAP WUN SHE

B.Eng. (Hons), Multimedia University, Malaysia

M. Eng. Sc., Multimedia University, Malaysia

THESIS SUBMITTED IN FULFILMENT OF THE

REQUIREMENT FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

(by Research)

in the

Faculty of Information Science and Technology

MULTIMEDIA UNIVERSITY
MALAYSIA

August 2015

DECLARATION

I hereby declare that the work has been done by myself and no portion of the work contained in this Thesis has been submitted in support of any application for any other degree or qualification on this or any other university or institution of learning.

_____

Yap Wun She

ACKNOWLEDGEMENTS

First and foremost, I thank my supervisor Prof. Dr. Heng Swee Huay and my co-supervisor Assoc. Prof. Dr. Ong Thian Song for giving me the chance to pursue my doctoral studies under their guidance. Everytime I finished a manuscript, Prof. Dr. Heng would give me detailed comments on it, both editorial and technical, which not only benefited my research, but also improved my editorial work. I am truly indebted to Prof. Dr. Heng as she (other than Prof. Dr. Goi Bok Min) initiated me into the field of cryptography during my master studies. Over the years, her valuable comments and constructive criticisms have been crucial in developing me as an independent researcher.

I thank my co-authors Jiqiang Lu, Sze Ling Yeo, Matt Henricksen, Chee Hoo Yian, Yongzhuang Wei, Bok-Min Goi, Raphael C.-W Phan and Wei-Chuen Yau for many fruitful discussions during my doctoral studies. I am totally amazed by Jiqiang on his passion and focus on research. His personal view on the research of symmetric key cryptography had inspired me to study on more important and widely used ciphers. Besides, I am motivated by Sze Ling. I am really grateful that I met and known Jiqiang and Sze Ling personally during my working experiences in Institute for Infocomm Research, Singapore. To me, both Jiqiang and Sze Ling are my unofficial co-supervisors who have helped to co-supervise me for most of my work during my Ph.D. study.

I thank my ex-colleagues and friends for the happy time spent together and the help provided, this includes Chunhua Su, Jian Guo, Joseph Liu, Aldar Chan, Jun-Wen Wong, Shen-Tat Goh, Jin Zhe, Syh-Yuan Tan, Huo-Chong Ling and Ji-Jian Chin. I also thank Lim Lian Tze for providing the LaTex template which is compatible with the format of Multimedia University postgraduate thesis.

Finally, my wife Soo Yee deserves special thanks for her support and her time to take care our lovely son Yi Tong and daughter Loo Xin. My wife has experienced every moment of my happiness and sadness through the bumpy road of my life.

To my parents, my wife, my son and my daughter.

ABSTRACT

This thesis contributes to the security analysis of block ciphers and block cipher based constructions which include message authentication codes, block cipher modes of operation and image encryption schemes.

First, we present the best cryptanalytic results on two block ciphers, namely, MISTY1 and SEED ciphers. These two block ciphers are the International Standardization of Organization (ISO) standards. More importantly, our results show that the MISTY1 cipher is distinguishable from an ideal cipher and thus cannot be regarded as an ideal cipher. We also present the first known cryptanalytic attack against the full CHAIN cipher based on a generalised impossible differential technique.

Next, we analyse both parallelisable message authentication code (PMAC) and Galois/counter mode (GCM) against forgery and distinguishing attacks. PMAC is part of the offset codebook (OCB) mode. Both OCB and GCM are ISO standards for mode of operation and recommended by National Institute of Standards and Technology. More importantly, the attack techniques developed for GCM can be applied to Wegmen-Carter polynomial message authentication codes and counter mode encryption. Our analysis on PMAC highlights some pitfalls that designers should be mindful of when designing cryptographic schemes which exploit the same design component, specifically the constant generation method.

Finally, in the context of image encryption schemes, we emphasise the importance of designing a secure key schedule algorithm by showing the weakness of such algorithms proposed in two recent image encryption schemes. Further, we disprove a general assumption that an image encryption scheme is secure against differential attack based on statistical tests by presenting the first impossible differential attack on an image encryption scheme that employs such a design rule. The results presented suggest that the designers should be mindful of widely studied cryptanalytic methods when designing a secure image encryption scheme.

TABLE OF CONTENTS

COPYRIGHT PAGE ii

DECLARATION iii

ACKNOWLEDGEMENTS iv

DEDICATION v

ABSTRACT vi

TABLE OF CONTENTS vii

LIST OF TABLES xi

LIST OF FIGURES xii

LIST OF ABBREVIATIONS xv

CHAPTER 1: INTRODUCTION 1

1.1 Problem Statement 1
1.2 Objectives 3
1.3 Contributions 3
1.4 Organisation of the Thesis 5

CHAPTER 2: PRELIMINARIES 7

2.1 Block Cipher 7
2.1.1 Descriptions 7
2.1.2 Two Common Design Methods 8
2.1.3 Security of Block Cipher 10
2.2 Applications of Block Cipher 28
2.2.1 Message Authentication Code 28
2.2.2 Authenticated Encryption/Mode of Operation 32
2.2.3 Image Encryption 33
2.3 Notation 34

CHAPTER 3: WEAK KEYS OF THE FULL MISTY1 BLOCK CIPHER FOR RELATED-KEY CRYPTANALYSIS 36

3.1 Introduction 37
3.2 The MISTY1 Block Cipher 41
3.3 A Related-Key Differential Attack of the Full MISTY1 under $2^{103.57}$ Weak Keys 43
3.3.1 A Class of $2^{102.57}$ Weak Keys Owing to Dai and Chen 44
3.3.2 Dai and Chen's 7-Round Related-Key Differential 46
3.3.3 Attacking the Full MISTY1 under the Class of $2^{102.57}$ Weak Keys 46
3.3.4 Another Class of $2^{102.57}$ Weak Keys 55
3.4 A Related-Key Amplified Boomerang Attack of the Full MISTY1 under 2^{92} Weak Keys 56
3.4.1 A Class of 2^{90} Weak Keys Owing to Chen and Dai 56
3.4.2 Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher 59
3.4.3 An Improved 7-Round Related-Key Amplified Boomerang Distinguisher 59
3.4.4 Attacking the Full MISTY1 under the Class of 2^{90} Weak Keys 64
3.4.5 Three Other Classes of 2^{90} Weak Keys 71
3.5 Summary 72

CHAPTER 4: DIFFERENTIAL ATTACK ON NINE ROUNDS OF THE SEED BLOCK CIPHER 73

4.1 Introduction 73
4.2 The SEED Block Cipher 75
4.3 Seven-Round Differentials of SEED 78
4.3.1 Sung's 7-Round Differentials 78
4.3.2 Our 7-Round Differentials 78
4.4 Differential Attack on 9-Round SEED 81
4.4.1 Precomputation 82
4.4.2 Attack Procedure 82
4.4.3 Attack Complexity 85
4.4.4 Notes 87
4.5 Summary 90

CHAPTER 5: IMPOSSIBLE DIFFERENTIAL ATTACK ON THE FULL CHAIN BLOCK CIPHER 91

5.1 Introduction 91
5.2 The CHAIN Block Cipher 93
5.3 On the Peyravian and Coppersmith's Differential Attacks 95
5.4 A Generic Impossible Differential Attack on $r+2$-Round CHAIN 98
5.4.1 Differential Characteristics with the Probability of One 98
5.5 Impossible Differential Attacks on CHAIN-64 and CHAIN-128 103

5.5.1 Impossible Differential Attack on 8-Round CHAIN-128 103
5.5.2 Impossible Differential Attack on 7-Round CHAIN-64 111
5.6 Summary 120

CHAPTER 6: GCM REVISITED 121

6.1 Introduction 122
6.1.1 Background 122
6.1.2 Related Work 123
6.1.3 Motivation and Contributions 125
6.2 The Galois/Counter Mode (GCM) 129
6.3 Weak Keys for GMAC 132
6.4 Universal Forgery on GMAC 135
6.5 Distinguishing Attack on GMAC 137
6.6 Known Ciphertext Attack on GCM 138
6.7 Summary 144

CHAPTER 7: PMAC REVISITED 146

7.1 Introduction 146
7.1.1 Related Work 147
7.1.2 Motivation and Contributions 149
7.2 The Parallelisable MAC (PMAC) 151
7.2.1 The Generation of Constants for PMAC1 153
7.2.2 The Generation of Constants for PMAC2 155
7.3 On the Security of PMAC1 156
7.3.1 Constructing Forgeries based on the Structure of the Gray Code 156
7.3.2 A General Birthday Attack on PMAC1 160
7.3.3 Almost Universal Forgery Attack on PMAC1 164
7.4 On the Security of PMAC2 165
7.4.1 Determining the Size of a and b 167
7.5 Discussion 169
7.6 Summary 171

CHAPTER 8: WEAKNESSES IN THE KEY SCHEDULE ALGORITHM OF RECENT IMAGE ENCRYPTION SCHEMES USING EXTERNAL KEY 172

8.1 Introduction 172
8.2 The `Key Schedule` Algorithm for Different Image Encryption Schemes 175
8.2.1 The `Key Schedule` Algorithm Proposed by X. Wang and Wang 175
8.2.2 The `Key Schedule` Algorithm Proposed by Norouzi and Mirzakuchaki 176
8.3 On the Effective Key Space of the `Key Schedule` Algorithm 177
8.3.1 The X. Wang and Wang Image Encryption Scheme 177

8.3.2 The Norouzi and Mirzakuchaki Image Encryption Scheme 181
8.4 Summary 183

CHAPTER 9: CRYPTANALYSIS OF A NEW IMAGE ALTERNATE ENCRYPTION ALGORITHM BASED ON CHAOTIC MAP 184

9.1 Introduction 184
9.2 The X. Wang and Guo Image Encryption Scheme 186
9.2.1 The `Key Schedule` Algorithm 186
9.2.2 The `E` Algorithm 187
9.3 Revisiting the Key Space of X. Wang and Guo Image Encryption Scheme 189
9.4 The Impossible Differential Attack on X. Wang and Guo Image Encryption Scheme 190
9.4.1 The *i*-Round Differential with F 190
9.4.2 The Impossible Differential Attack 193
9.5 A Divide-and-Conquer Attack using A Large All Black Image 197
9.6 Summary 198

CHAPTER 10: CONCLUSION AND FUTURE WORK 200

10.1 Conclusion 200
10.2 Future Work 202

REFERENCES 204

LIST OF PUBLICATIONS 223

LIST OF TABLES

Table 3.1	Main Cryptanalytic Results on MISTY1 with FL Functions	40
Table 4.1	Main Cryptanalytic Results of Differential Cryptanalysis on SEED	75
Table 4.2	Our 7-Round Differentials of SEED	81
Table 5.1	Main Cryptanalytic Results of Impossible Differential Cryptanalysis on CHAIN	92
Table 6.1	The Complexities of the Various Attacks on GMAC	128
Table 7.1	Number of Tagging Queries for Different n-Bit Block Sizes and Message Blocks in Each Query	163
Table 7.2	The Values of u for the Various Primitive Irreducible Polynomials of Degree 8	168
Table 7.3	Discrete Log Values for Different Irreducible Polynomials	169
Table 8.1	The Effective Key Space of Two Image Encryption Schemes	174
Table 8.2	Compute Q_1 to Q_{32}	176
Table 8.3	$\#(K_{26}, K_7, sum)$ that Results d_1	178
Table 8.4	256 Possible Values of b	179

LIST OF FIGURES

Figure 2.1	Two General Design Methods to Construct a Block Cipher	9
Figure 2.2	A Double-Encryption Scheme	16
Figure 2.3	The MITM Attack on a Double-Encryption Scheme	17
Figure 2.4	Impossible Differential: Miss-In-The-Middle Approach	21
Figure 2.5	The Boomerang, Amplified Boomerang and Rectangle Distinguishers	24
Figure 2.6	The Related-Key Amplified Boomerang Distinguisher	27
Figure 3.1	MISTY1 and Its Components	42
Figure 3.2	Chen and Dai's Related-Key Differential for Rounds 2–4	47
Figure 3.3	Chen and Dai's Related-Key Differential for Rounds 5–8	48
Figure 3.4	Propagation of α through the Inverse of Round 1 with $\mathbf{FL}_1$ and $\mathbf{FL}_2$	51
Figure 3.5	Thet Related-key Differential for Rounds 1-2 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher	60
Figure 3.6	Thet Related-key Differential for Rounds 3-4 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher	61
Figure 3.7	Thet Related-key Differential for Rounds 5-7 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher	62
Figure 3.8	Propagation of δ Through Round 8 with $\mathbf{FL}_9$ and $\mathbf{FL}_{10}$	66
Figure 4.1	The Feistel Structure of the SEED Block Cipher	76
Figure 4.2	The Structures of the **F** and **G** Functions	76
Figure 4.3	7-Round Differentials of SEED	79
Figure 4.4	Differential Attack on 9-Round SEED	83

Figure 5.1 The Round Function **F** of CHAIN 94

Figure 5.2 The 2-Round Iterated Differential Characteristic of CHAIN 97

Figure 5.3 The 4-Round Differential Characteristic S on CHAIN-128 with Probability of One 104

Figure 5.4 The 2-Round Differential Characteristic T on CHAIN-128 with Probability of One 105

Figure 5.5 The Impossible Differential Attack on 8-Round CHAIN-128 106

Figure 5.6 One-Round Decryption of Round 8 of CHAIN-128 108

Figure 5.7 The 3-Round Differential Characteristic S with Probability of One 112

Figure 5.8 The 2-Round Differential Characteristic T with Probability of One 113

Figure 5.9 The Impossible Differential Attack on 7-Round CHAIN-64 115

Figure 5.10 1-Round Decryption of Round 7 of CHAIN-64 116

Figure 6.1 The GCTR Function 131

Figure 6.2 The Polynomial Hash GHASH 132

Figure 6.3 The Theoretical Proportion of the Size of Set D for Different Values of k 142

Figure 6.4 The Average Proportion of the Size of Set D for Different Values of k 143

Figure 7.1 Overall Structure of PMAC 152

Figure 7.2 Diagram of the General Birthday Attack on PMAC1 161

Figure 8.1 The MITM Attack on X. Wang and Wang Image Encryption Scheme 180

Figure 9.1 Graphical Presentation of **Encrypt** with the Number of Rounds r 188

Figure 9.2 The *i*-Round Differential of the X. Wang and Guo Image Encryption Scheme with Probability of One 191

LIST OF ABBREVIATIONS

AES Advanced Encryption Standard.

CMAC Cipher-based Message Authentication Code.

DDT Difference Distribution Table.

ESP Encapsulating Security Payload.

GCM Galois/Counter Mode.

IETF Internet Engineering Task Force.

ISO the International Standardization of Organization.

MAC Message Authentication Code.

MITM Meet-In-The-Middle.

NIST National Institute of Standards and Technology.

NPCR Number of Pixels Change Rate.

OCB Offset Codebook Mode.

PMAC Parallelisable Message Authentication Code.

UACI Unified Average Changing Intensity.

UMTS Universal Mobile Telecommunication System.

CHAPTER 1

INTRODUCTION

In this introductory chapter, we first motivate the research undertaken for this thesis. We then describe the contributions of our research and finally present the overall structure of this thesis.

1.1 Problem Statement

Due to the rapid advancement in technology, governments over the world increasingly tap on opportunities provided by the next generation infrastructure with a particular focus on coverage, affordability and quality of the access of broadband. The improved broadband bandwidth speed at a cheaper cost enables the massive transmission of multimedia data between users more frequently. Multimedia data include text, voice and image. In order to provide confidentiality and authenticity of data transmitted over insecure communication channels at high speed, symmetric key cryptographic primitives are frequently employed in existing security applications for communication purposes.

Indeed, symmetric key cryptographic primitives are frequently employed in existing security applications for communication purposes due to their superior efficiency in terms of speed as compared to public key cryptographic primitives. Out of all symmetric key cryptographic primitives, the block cipher is the most widely used symmetric key primitive in real-life applications as the block cipher can be used as the core component in building other symmetric key primitives such as the stream cipher, hash function, message authentication code (MAC) and modes of operation. Some commonly-used block ciphers and block cipher based constructions include KeeLoq block cipher used in wireless devices that unlock the doors and alarms in cars manufactured by a number of automotive companies, such as Honda, Jaguar, Toyota, Volvo,

Volkswagen, etc (Courtois, Bard, & Wagner, 2008), KASUMI block cipher (3GPP, 2001) (which was designed based on MISTY1 block cipher (Matsui, 1997)) used in Universal Mobile Telecommunication System (UMTS) to provide both confidentiality and integrity services, SEED block cipher (KISA, 1998) used by Mozilla Firefox web browser as a cipher to provide confidentiality service in Transport Layer Security (TLS) protocol, Galois/counter mode (GCM) used in the IEEE MAC standard, TLS protocol and Internet Engineering Task Force (IETF) Internet Protocol Security, offset codebook mode (OCB) used in IEEE wireless security standard and cipher-based message authentication code (CMAC) (Dworkin, 2001) used in IPsec Encapsulating Security Payload (ESP).

In order to design a block cipher which is efficient in terms of speed, designers propose the use of simple operations, such as exclusive-or, modular addition, modular multiplication, bitwise shift, bit-permutation and bitwise rotation operations. As simple operations are used, it is common that no security proofs are given in linking the security of a block cipher with a mathematical hard problem. When such block cipher is used in constructing different primitives such as MAC and authenticated encryption scheme, designers construct the security proofs of their proposed block cipher based primitives based on the fact that the underlying block cipher is a random permutation even though that one cannot assure the underlying block cipher is indeed a random permutation. Generally, designers evaluate the security of their proposed block cipher against various conventional cryptanalytic methods (e.g., differential-like and linear attacks) based on their personal experiences due to the high computation power needed in searching a distinguisher with non-negligible probability. To boost the confidence of public on the proposed block cipher, it is important for the third party to perform additional security analysis on the security of block ciphers and block cipher based constructions as such analysis serves as the certificate of its security strength and the pitfalls that designers should be mindful of when designing a block cipher and a block cipher based construction.

1.2 Objectives

This thesis is concerned with the cryptanalysis of block ciphers and block cipher based constructions which include the message authentication code, mode of operation and image encryption scheme. The background of block ciphers and their constructions will be provided in Chapter 2. To be precise, the three main objectives of this research are listed as follows:

1. To find better differentials under different assumptions of secret key (i.e., weak key, single key and related-key) against different block ciphers and block cipher based constructions in terms of greater number of round and/or greater probability.
2. To exploit the structural properties of block ciphers and block cipher based constructions in attacking such primitives.
3. To critique existing statistical tests based design rules of image encryption schemes from a cryptanalytic point of view.

1.3 Contributions

Our contributions in this thesis are listed as follows:

- **Cryptanalysis of block ciphers.** The MISTY1 and SEED block ciphers have received considerable attention since their publication and their security has been thoroughly analysed as both MISTY1 and SEED block ciphers are the International Standardization of Organization (ISO) standards for block cipher. We present related-key differential and related-key amplified boomerang attacks on the full MISTY1 under certain weak key assumptions. Our results are the first to exhibit a cryptographic weakness in the full MISTY1 cipher. More importantly, we show that the MISTY1 cipher is distinguishable from an ideal cipher and thus cannot be regarded as an ideal cipher. Besides, we describe two 7-round differentials with probabilities that are slightly larger than the best previously known

one on SEED and present a differential cryptanalysis attack on a 9-round reduced version of SEED. For SEED cipher, our result is better than any previously published cryptanalytic results on SEED in terms of the number of attacked rounds and it suggests for the first time that the safety margin of SEED decreases below half of the number of rounds. Lastly, we present an impossible differential attack on full CHAIN block cipher based on a generalised impossible differential. To the best of our knowledge, this is the first known cryptanalytic attack against CHAIN which is a block cipher with a variable block length, a variable secret key length and a variable number of rounds proposed by Peyravian and Coppersmith (1999) from IBM.

- **Cryptanalysis of message authentication codes and block cipher modes of operation.** We perform the security analysis on the GCM and the parallelisable message authentication code (PMAC). The result is interesting as GCM is an ISO standard for mode of operation while PMAC is a part of the offset codebook mode (OCB) which is an ISO standard for mode of operation too. For GCM, it combines both counter mode encryption GCTR and a message authentication code GMAC to provide both confidentiality and authenticity. We analyse the security of GMAC and GCM with respect to the forgery and distinguishing attacks. The attack techniques can be applied to other Wegmen-Carter polynomial message authentication codes (Wegman & Carter, 1981). We remark that part of the results were independently found by Procter and Cid (2013). In addition, we can utilise the uniqueness of the counter mode encryption to launch a known ciphertext attack against GCM itself. Such uniqueness of the counter mode encryption gives rise to a technique referred to as an impossible plaintext cryptanalysis. This technique was named and independently found by McGrew (2012). With regards to PMAC, there exists different version of PMAC that use different constant generation methods. We analyse how some unique characteristics of these constants result in weaknesses of the respective PMAC designs against forgery attacks in different ways. Thus, our analysis highlights some pitfalls that designers should be mindful of when designing schemes which employ such constants.

- **Cryptanalysis of image encryption schemes.** We investigate the security of chaos based image encryption schemes. A compulsory condition for the security of an image encryption scheme is that the length of the external secret key should be sufficiently long in terms of bit length. However, the sufficiently long secret key is not a guarantee that the scheme is secure. In this thesis, we emphasise the importance of designing a secure `Key Schedule` algorithm by showing the weakness of the `Key Schedule` algorithms employed in two recent image encryption schemes proposed by X. Wang and Wang (2014) and by Norouzi and Mirzakuchaki (2014). Besides, an image encryption scheme is said to have good property against differential attack if the measured number of pixels change rate (NPCR) and unified average changing intensity (UACI) are close to the ideal values. However, we disprove such general assumptions by showing that these two values are not adequate to conclude that an image encryption scheme is secure against the impossible differential attack which is a type of differential-like attack. To the best of our knowledge, we present the first impossible differential attack on an image encryption scheme proposed by X. Wang and Guo (2014). The results suggest that an image encryption scheme must be designed to withstand widely studied cryptanalytic methods. Most importantly, it suggests that the current ground rule to design an image encryption scheme is inadequate to result in a secure image encryption scheme.

1.4 Organisation of the Thesis

The remainder of this thesis is organised as follows.

- **Preliminaries.** In Chapter 2, we provide the background that is needed to understand our results in the remainder of this thesis. We describe a block cipher and its applications to construct message authentication codes, authenticated encryptions (also known as block cipher mode of operation) and image encryptions. Our results focus on analysing the security of block ciphers and its applications, and thus we describe how a cryptographic design can be considered as a secure design. In order to understand what an attack involves, we briefly describe the re-

sources needed to quantify a cryptanalytic attack. Subsequently, we summarise the commonly used cryptanalytic methods to attack block ciphers.

- **Our new cryptanalytic results.** In Chapter 3 to 9, we present our cryptanalytic results on MISTY1 (Matsui, 1997), SEED (KISA, 1998), CHAIN (Peyravian & Coppersmith, 1999), GCM (McGrew & Viega, 2005), PMAC (Black & Rogaway, 2002; Rogaway, 2004) and some image encryption schemes (Norouzi & Mirzakuchaki, 2014; X. Wang & Guo, 2014; X. Wang & Wang, 2014) respectively. In each chapter, we begin with an introduction and the specification of the block cipher or the block cipher based construction in question. We then proceed to present the cryptanalytic results on these symmetric key primitives. Finally, we summarise our findings.

- **Conclusion and future work.** In Chapter 10, we summarise the results obtained in this thesis and conclude the thesis with some suggestions for future research work.

CHAPTER 2

PRELIMINARIES

In this chapter, we provide the background needed to understand our results in the remainder of this thesis. In particular, we describe a block cipher and its applications to construct message authentication codes, authenticated encryption schemes and image encryption schemes. As our results focus on the security analysis of block ciphers and their applications, we describe how a cryptographic design can be considered as secure. In order to understand what an attack involves, we briefly explain the resources needed to quantify a cryptanalytic attack. Subsequently, we provide the commonly used cryptanalytic methods to attack block ciphers.

2.1 Block Cipher

2.1.1 Descriptions

As its name suggests, a block cipher is a cipher that processes an n-bit input block to generate an n-bit output block under a k-bit secret key K. The input (respectively the output) is commonly termed the plaintext P (respectively the ciphertext C). Typically, n is 64-, 80- or 128-bit and k is 80-,128-,192- or 256-bit. To be precise, a block cipher is a cipher that consists of the following three algorithms:

1. `Key Schedule`: This algorithm derives i-round subkeys rk_i for $i > 0$ under the control of a k-bit secret key K.

2. `E`: This algorithm encrypts an n-bit plaintext P to an n-bit ciphertext C determined by the subkeys. Such an operation is denoted as $C = \mathtt{E}_K(P)$.

3. `D`: This algorithm decrypts an n-bit ciphertext C to an n-bit plaintext P determined by the subkeys. Such an operation is denoted as $P = \mathtt{D}_K(C)$.

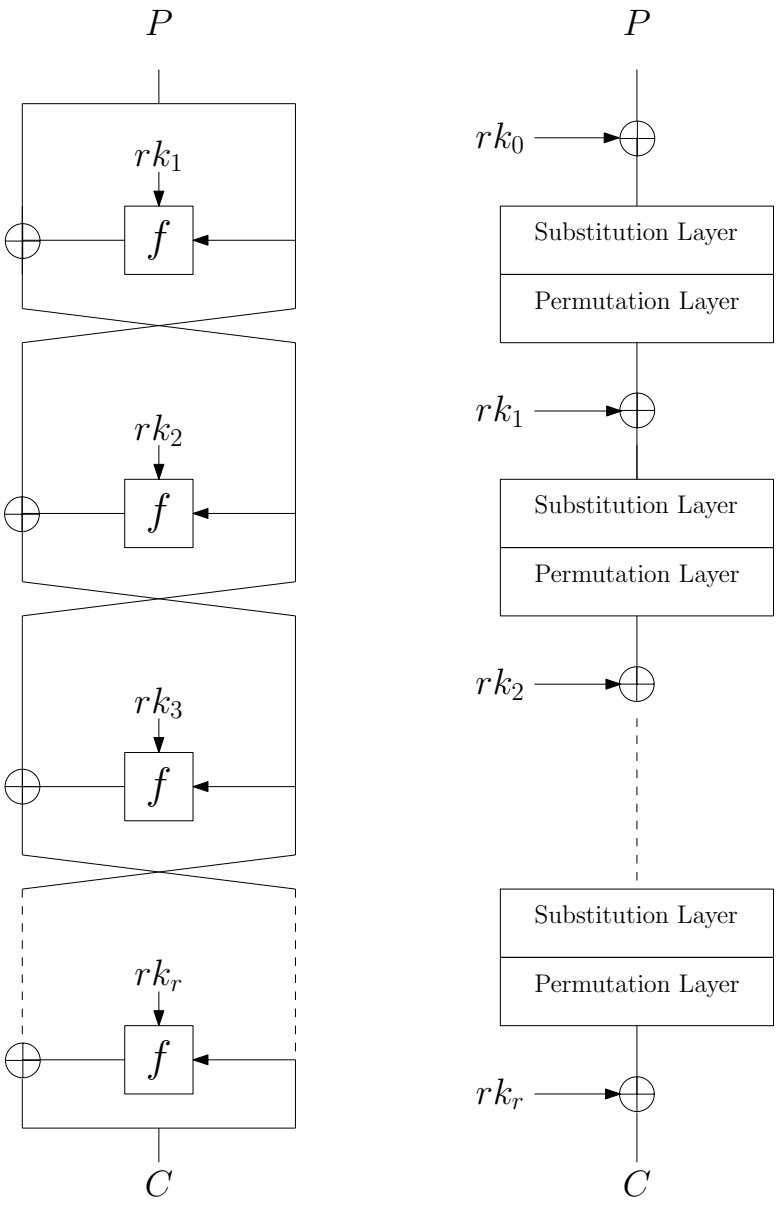
It is obvious that D is the inverse of E given that the same secret key is used for both algorithms. Throughout this thesis, we denote n as the block size and k as the length of the secret key K in bits.

2.1.2 Two Common Design Methods

The rule of thumb to design a secure block cipher is to design a weaker round or internal function f that can provide adequate confusion and diffusion of the bits (Shannon, 1949). Confusion means that each bit of the ciphertext should be affected by as many bits of the secret key as possible while diffusion means that changing a bit of the plaintext (respectively ciphertext) will change a large number of bits of the ciphertext (respectively plaintext). Further, the round function is iterated a number of times to form a cryptographically strong function that can encrypt a plaintext to a ciphertext. Each iteration is known as a round and the number of iterations is called the number of rounds r of the cipher. Such a design method forms an iterated cipher.

The two main designs to construct a block cipher are the Feistel network (FN) and substitution-permutation network (SPN) as shown in Figure 2.1.

- Feistel network: The Feistel cipher was named after the cryptographer Horst Feistel who designed the Lucifier block cipher (Feistel, 1970, 1973), a predecessor of Data Encryption Standard (DES) (NBS, 1977) which is by far the most popular and most widely implemented block cipher. A Feistel cipher first splits the plaintext into two halves, specifically left half and right half. A subkey dependent round function f is then applied on the right half and the output of f will be exclusive-ored with the left half. Lastly, both halves are swapped and this will form the input to the next round. This process will be iterated for r times. To have the same E and D algorithms, both halves will not be swapped in the last round. Since the same algorithm can be used for both encryption and decryption (note that the subkeys are applied in reverse order for one of them), this leads to a smaller code/circuitry size to implement a Feistel cipher. Early block ciphers



a) A Feistel Cipher b) A SPN Cipher

Figure 2.1: Two General Design Methods to Construct a Block Cipher

were mainly designed based on the Feistel Network's principle. This includes a number of block ciphers specified by ISO, such as Camellia (Aoki et al., 2001), CAST-128 (Adams, 1997), MISTY1 (Matsui, 1997), SEED (KISA, 1998) and Triple DES (ISO, 2010).

- Substitution-permutation-network: Advanced Encryption Standard (AES) (NIST, 2001) proposed by Rijmen and Daeman is a de facto cipher designed based on substitution-permutation-network. An SPN cipher first exclusive-ored the plaintext with a subkey for the purpose of pre-whitening. Similar to a Feistel cipher, a round function f will then be applied to the data for r rounds. In each round function f, the data is fed into the substitution layer and permutation layer sequentially. The $\mathtt{E}$ algorithm differs from the $\mathtt{D}$ algorithm where the inverses of the substitution layer and permutation layer must be designed in order to decrypt the ciphertext to the right plaintext. The main advantage of a SPN cipher is that all the bits of the data is changed in each round as compared to the Feistel cipher which only changes half of the bits in each round. Besides, the inherent parallelism is another clear advantage. Since all the bits of the data is changed in each round, an SPN cipher typically consists of fewer rounds as compared to a Feistel cipher and this leads to faster encryption. Another ISO standard for block cipher that is constructed based on the principle of substitution-permutation-network is HIGHT (D. Hong et al., 2006).

In order to achieve faster encryption speed, block ciphers are designed using simple operations, such as exclusive-or operation (denoted by $\oplus$ or XOR), modulo addition (denoted by $\boxplus$), bitwise rotation, bitwise shift operation, bit permutation and substitution box (S-box). We denote by $i \times j$-bit S-box as an S-box with an i-bit input and j-bit output. If $i = j$, we call it an i-bit S-box for simplicity throughout this thesis.

2.1.3 Security of Block Cipher

In order to demonstrate that the designed construction is secure, designers need to provide a security proof which ensures that such a scheme is secure under an ac-

ceptable security model which specifies the attacker abilities. However, the priority of a block cipher designer is to construct a block cipher that is elegant and simple to achieve a practical encryption speed. Thus, simple operations have been utilised to achieve such aims. To the best of our knowledge, none of the commonly-used international standardised ciphers has been proven secure with a rigorous security proof. In general, the design of a block cipher is ad hoc, and the designers will evaluate its security against existing cryptanalytic methods. Block ciphers are thus always endangered by the possibility of new cryptanalytic methods.

According to Shannon (1949), the father of information theory, a cryptographic primitive should be designed to achieve either unconditional security or computational security. The distinction between these two types of security is the computational power of an attacker. A primitive is unconditionally secure if it is unbreakable even though the attacker has unlimited computational power. On the other hand, a primitive is computationally secure if it is unbreakable in reasonable time when the attacker has limited computational power. Clearly, unconditional security is ideal for any constructed cryptographic primitives, but it is not practical to achieve such unconditional security. For example, the one-time pad is unconditionally secure when the random generated key K is as long as the plaintext. Nonetheless, it is not practical for a user to memorise/store such a long secret key given that the plaintext encrypted is usually as big as few gigabytes. This gives rise to the design of a cryptographic primitive which is computationally secure.

2.1.3 (a) Classification of Cryptographic Attacks

Since computational security is our main concern, we consider an attack feasible if the resources to launch this attack are reasonable. Before describing the resources that quantify an attack, we first explain the hierarchy of an attacker's goals in descending order as follows:

1. Total break: The attacker successfully recovers the user secret key.

2. Global deduction: The attacker successfully finds an algorithm that is functionally equivalent to either $\mathtt{E}(\cdot)$ or $\mathtt{D}(\cdot)$.

3. Local deduction: The attacker can recover the plaintext (respectively ciphertext) corresponding to a new ciphertext (respectively plaintext).

4. Distinguishing algorithm: The attacker can effectively distinguish whether a black box contains a block cipher with a random chosen secret key or a random chosen permutation.

Note that a cipher cannot be treated as an ideal cipher if it is vulnerable to the distinguishing attack. Such a non-ideal cipher is not suitable for any security application as security proofs are constructed based on the assumption that the underlying cipher is a random chosen permutation.

The effectiveness of an attack is judged using three metrics that indicate the amount of available resources as follows:

1. Time: This can be considered as the most important metric. The unit of time is measured in terms of the number of encryptions. An attack is practical under current technology if the time complexity of such an attack is around 2^{64} encryptions. The current minimum acceptable security level is of 80-bits, namely, no attack with a time complexity smaller than 2^{80} encryptions should be available.

2. Memory/Storage: An attack may require extra storage (be it a hard disk or random-access memory) to store partial/intermediate results. An attack is impractical when the amount of memory required is too large. Generally, it is more difficult to launch an attack that needs higher memory complexity as compared to time complexity. For instance, it is more practical to run an attack that requires 2^{50} encryptions as compared to an attack that requires a memory size of 2^{50} bits.

3. Data: Besides time and memory, an attack may require certain amount of data to break a cryptographic primitive. The impact of an attack is limited when the amount of data to launch such an attack is too large as it is logical to assume that the attacker can only obtain limited data in practice.

The type of data available to an attacker is important as well. Indeed, not all data are useful or available to the attacker. Generally, an attack can be classified according to the different types of data needed as follows (Biryukov, 1999; Knudsen & Robshaw, 2011; Lu, 2008):

1. Ciphertext only: This is the most realistic scenario as the attacker can only get access to a number of ciphertexts by eavesdropping the communication channel between a sender and a receiver. The attacker is assumed to have minimal knowledge of the plaintexts, such as the written format and language used to generate them.

2. Known plaintext: The attacker is assumed to have access to a number of ciphertexts and the corresponding plaintexts. This scenario is still highly realistic.

3. Chosen plaintext (respectively ciphertext): The attacker is assumed to have access to an $\mathtt{E}$ (respectively $\mathtt{D}$) oracle where the attacker has the ability to generate the ciphertext (respectively plaintext) of his chosen plaintext (respectively ciphertext).

4. Adaptive chosen plaintext (respectively ciphertext): Similar to the above chosen plaintext (respectively ciphertext) scenario, the attacker is assumed to have access to an $\mathtt{E}$ (respectively $\mathtt{D}$) oracle. Moreover, the attacker has the ability to obtain the ciphertext (respectively plaintext) of his chosen plaintext (respectively ciphertext) based on his previous queried chosen plaintext-ciphertext pairs.

In the above attacks, the different plaintexts (respectively ciphertexts) are assumed to be obtained from a $\mathtt{D}$ (respectively $\mathtt{E}$) algorithm under a single unknown

secret key. This assumption of the key is realistic as the user may use a same single unknown secret key for encryption and decryption over his lifetime. A less realistic assumption of the key is known as related-key assumption. Under the related-key scenario, the attacker may obtain the plaintexts (respectively ciphertext) decrypted (respectively encrypted) from a $\mathtt{D}$ (respectively $\mathtt{E}$) algorithm under different unknown secret keys and the attacker knows the relationship between one or more pairs of such unknown secret keys. Certain current real-world applications may allow for practical related-key attacks, for example, the key-exchange protocol (Kelsey, Schneier, & Wagner, 1996).

Last but not least, to evaluate the security of any cryptographic primitives, the attacker is assumed to have full knowledge of the construction of such cryptographic primitives except the unknown secret key. This assumption is called the Kerckhoffs's principle (Kerckhoffs, 1883). This assumption is used in practice as it will definitely raise the confidence of the public on the security of their designed primitives if no weaknesses can be found against their design even though all details are made public.

2.1.3 (b) Cryptanalytic Methods

A cryptanalyst will play the role of an attacker to seek for weaknesses in a cryptographic primitive. Even though such weaknesses may not pose a direct and practical threat to the use of the primitive, say the time complexity exceeds 2^{80} but smaller than 2^k where the secret key K is of k-bit long, the existence of such vulnerabilities remain interesting, especially from a theoretical point of view.

Over the last 25 years, a number of cryptanalytic methods had been developed to evaluate the computational security of a block cipher assuming the attacker has a bounded computational power. The methods used to break a cipher can be classified into two categories, namely, generic attacks and cryptanalytic attacks.

1. Exhaustive key search: Exhaustive key search is often used interchangeably with brute-force attack. In an exhaustive key search attack, given a known plaintext-ciphertext pair (P,C), the attacker simply tries every single possible key K' and check whether $C' \stackrel{?}{=} C$ where $C' = \mathtt{E}_{K'}(P)$. Observe that the number of guessed keys that survives this checks is approximately 2^{k-n}. If $k > n$, more than one key remain and thus, additional known plaintext-ciphertext pairs are needed to filter out all the remaining wrong keys to obtain the correct key. Generally, such a brute-force attack has a time complexity of 2^k encryptions and negligible data and memory complexities.

2. Dictionary attack: In a dictionary attack, the attacker first determines a widely transmitted plaintext P between a sender and a receiver. Subsequently, the attacker will encrypt such plaintext P under every single possible secret key. Those generated ciphertexts are then stored in a memory lookup table along with the corresponding guessed secret key. The generation of such a table has a time complexity of 2^k encryptions and a memory complexity of 2^k n-bits. Even though the complexities are large, the attacker can perform these computations offline. These complexities are considered as pre-computation complexities. In practice, the attacker will eavesdrop the communication channel between a sender and a receiver to intercept the ciphertext C of the plaintext P. The obtained ciphertext is then compared with those ciphertexts stored in the memory lookup table. To speed up the memory access operation, those ciphertexts are often stored in a hash table where a memory access requires $O(1)$ operations. Obviously, the online attack has a negligible time complexity.

3. Codebook attack: Instead of trying every possible secret key to encrypt a particular plaintext, the attacker tries to gather 2^n ciphertexts of all 2^n possible plaintexts under the same secret key. A table will be created to store the relationship between the plaintext and the ciphertext. Given a ciphertext C which is obtained from eavesdropping the communication channel between a sender and a receiver

(under the ciphertext-only scenario), the attacker can deduce the corresponding plaintext from the table provided C is generated using the E algorithm under the same secret key.

4. Birthday attack: The birthday paradox finds many applications in attacks on block ciphers, message authentication codes and cryptographic hash functions. The birthday attack makes use of the birthday paradox arguments to find a collision that can lead to the breaking of a cryptographic scheme. For example, given a function $y = f(x)$ where y is i-bit long, one may find a collision such that $f(x_1) = f(x_2)$ with a non-negligible probability if he can obtain $y_j = f(x_j)$ for $j = 1$ to $2^{i/2}$.

5. Meet-in-the-middle attack: The meet-in-the-middle (MITM) attack was first proposed by Diffie and Hellman (1977) to evaluate the security of multiple encryptions without considering the internal structure of a single encryption. A different secret key is used for each of the encryptions. To explain the MITM attack, we give an example using a double-encryption scheme as shown in Figure 2.2. Mathematically, we can write the ciphertext $C = \mathrm{E}_{K_2}(\mathrm{E}_{K_1}(P))$.

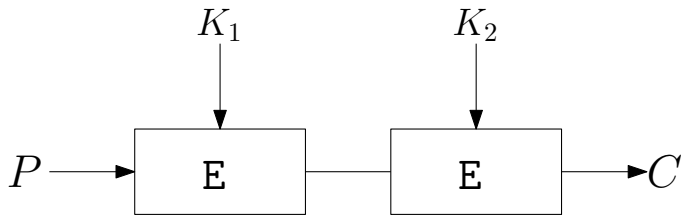


Figure 2.2: A Double-Encryption Scheme

Given a known plaintext-ciphertext pair (P,C), the MITM attack can be applied to the double-encryption scheme as follows. Figure 2.3 shows the graphical illustration of the MITM attack against a double-encryption scheme.

- For each of the 2^{k_1} k_1-bit secret keys K_1', compute and store $u = \mathrm{E}_{K_1'}(P)$ in a memory lookup table along with K_1'. This step has a time complexity of 2^{k_1} encryptions and such a table has a memory complexity of 2^{k_1} n-bits.

- For each of the 2^{k_2} secret keys k_2-bit K_2', compute $v = \mathrm{D}_{K_2'}(C)$ and compare v with u stored in the hash table. If they match, then (K_1', K_2') can be the right secret keys.

The number of possible secret key pairs (K_1', K_2') remaining after the above process is $2^{k_1+k_2-n}$. Assume that $k_1 = k_2 = k = n$. Then 2^k possible secret keys are left. One additional plaintext-ciphertext pair is needed to filter out all the remaining wrong keys except the right key using exhaustive key search approach. The overall time complexity of such an MITM attack is around $2^k + 2^k + 2^k \approx 3 \cdot 2^k$ encryptions. Even though a $2k$-bit secret key is used for a double-encryption scheme, such a scheme can only offer around k-bit security which implies that a double-encryption scheme is as secure as a single encryption scheme while its encryption speed is halved. Readers may refer to the papers (Bogdanov & Rechberger, 2011; Isobe, 2011, 2013) for MITM-related attacks after combining MITM with other techniques.

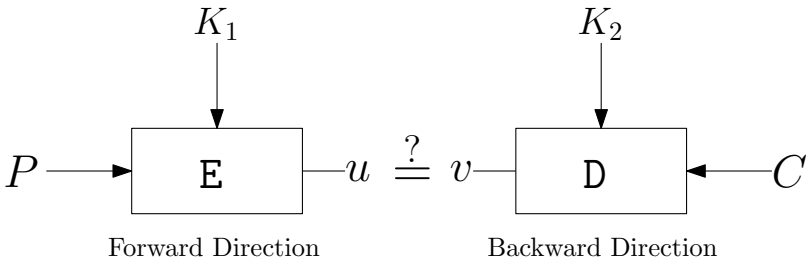


Figure 2.3: The MITM Attack on a Double-Encryption Scheme

Cryptanalytic Attacks

Next, we present some cryptanalytic attacks focusing on those employed in this thesis. In particular, we present a basic version of differential cryptanalysis and its various generalisations including impossible differential, boomerang, amplified boomerang, rectangle and related-key attacks.

In the following, the difference between pairs of inputs, outputs and keys are known as input difference, output difference and key difference respectively. Let X_i and X_i' denote the intermediate values which are the inputs of each round function i. The difference between the intermediate values is called the intermediate difference.

1. Differential attack: Differential attack was first proposed by Biham and Shamir (1991a, 1991b) and variants of the technique had been applied to attack DES-like ciphers and hash functions (Biham & Shamir, 1991c, 1993). This includes DES (NBS, 1977), FEAL (Shimizu & Miyaguchi, 1988), N-Hash (Miyaguchi, Ohta, & Iwata, 1990), Snefru (Merkle, 1990), Khafre (Merkle, 1991), REDOC-II (Cusick & Wood, 1991), LOKI (Brown, Pieprzyk, & Seberry, 1990) and Lucifier (Feistel, 1970, 1973).

   The principle of differential cryptanalysis is to study how a particular input difference can affect a particular output difference where the outputs are obtained by encrypting the inputs under a fixed unknown secret key. The difference can be an exclusive-or operation (denoted as $\oplus$ or XOR) or a modular addition difference (denoted as $+$ or $\boxplus$).

   A differential characteristic is a path that specifies the propagation from a particular input difference to a particular output difference. There may exists more than one differential characteristic with the same input and output differences but different intermediate differences. Joining all differential characteristics forms a differential.

   A differential (characteristic) for i consecutive rounds is commonly called an i-round differential (characteristic) while an i-round differential (characteristic) that has a probability p is called an i-round differential (characteristic) with probability p. As stated by Lu (2008), the probability of an i-round differential for a block cipher under a secret key K is defined as follows.

   Definition 2.1. *Let E be an n-bit block cipher and let $K \in \{0,1\}^k$ denote a secret key for E. If α and β are n-bit values, then the probability of the i-round*

differential (α, β) *for* E_K*, written as* $\Delta\alpha \xrightarrow{i} \Delta\beta$ *(or* $\alpha \xrightarrow{i} \beta$*), is defined as*

$$\Pr_{E_K}(\Delta\alpha \xrightarrow{i} \Delta\beta) = \Pr(E_K(P) \oplus E_K(P \oplus \alpha) = \beta).$$

Proposition 2.1 follows trivially from Definition 2.1.

Proposition 2.1. *Let* E *be an n-bit block cipher, and let* $K \in \{0,1\}^k$ *denote a secret key for* E*. For n-bit values* α *and* β*,*

$$\Pr_{E_K}(\Delta\alpha \xrightarrow{i} \Delta\beta) = \frac{\{\#x | E_K(x) \oplus E_K(x \oplus \alpha) = \beta, x \in \{0,1\}^n\}}{2^n}.$$

We usually refer to the differential (characteristic) of a block cipher without specifying the secret key as the differential (characteristic) probability does not depend on the secret key used for most of the existing block ciphers.

If an attacker finds a i-round differential $\Delta\alpha \xrightarrow{i} \Delta\beta$ with non-negligible probability p, the attacker can exploit such a differential to find the right subkeys by treating $\mathtt{E} : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ as a cascade of three sub-ciphers, that is, $\mathtt{E} = \mathtt{E}_a \circ \mathtt{E}_p \circ \mathtt{E}_b$ where $\mathtt{E}_p$ denotes the rounds for which $\Delta\alpha \xrightarrow{i} \Delta\beta$ holds, $\mathtt{E}_a$ denotes the rounds before $\mathtt{E}_p$ and $\mathtt{E}_b$ denotes the rounds after $\mathtt{E}_p$.

For each guess for the subkeys used in $\mathtt{E}_a$ and $\mathtt{E}_b$, the counter is incremented if an input difference produces a difference of α after $\mathtt{E}_a$ and its output difference produces a difference of β before $\mathtt{E}_b$. Thus, given a sufficient number of matching plaintext-ciphertext pairs, an attacker can deduce the correct subkeys involved in the number of rounds before and after $\mathtt{E}_p$ as the correct subkeys most probably are the guessed keys that are suggested with higher frequency.

Note that instead of determining the secret key via brute-force, the aim of the attacker is to reduce it to a brute-force search of partial subkeys involved in the number of rounds before and after $\mathtt{E}_p$. The complexity of the attack will be reduced since a round function is usually not designed to be cryptographically strong. Once the subkeys in the number of rounds before and after $\mathtt{E}_p$

are successfully recovered, the attacker can continue the same attack by peeling off the number of rounds before and after $\mathtt{E}_p$. Further, from the recovered subkeys, the attacker may be able to recover partial information of the secret key by analysing the `Key Schedule` algorithm. In such a scenario, brute-force attacks can be applied on the block cipher with the time complexity smaller than 2^k encryptions.

Differential attack was a major breakthrough in the area of cryptanalysis in 1990. Since then, a number of new cryptanalytic techniques inspired by the differential attack had been introduced, including higher-order differential cryptanalysis (Knudsen, 1995; Lai, 1994), impossible differential cryptanalysis (Biham, Biryukov, & Shamir, 1999a; Knudsen, 1998), boomerang and rectangle attacks (Biham, Dunkelman, & Keller, 2001; Kelsey, Kohno, & Schneier, 2001; Wagner, 1999), truncated differential cryptanalysis (Knudsen, 1995) and multiple differential cryptanalysis (Blondeau & Gérard, 2011).

2. Impossible differential attack: The impossible differential attack technique was first used by Knudsen (1998) to attack DEAL block cipher before Biham et al. (1999a) formalised the technique and named it the impossible differential attack. As the name implies, the main idea of impossible differential attack is to construct differentials that hold with probability of zero in order to discard those keys that lead to this impossible differential. Thus, the right key space can be significantly reduced using the aforementioned check. Note that the impossible differential attack is opposed to the conventional differential attack (Biham & Shamir, 1991a) where an attacker tries to seek for a differential with high probability p.

   The construction of an impossible differential can be achieved through the miss-in-the-middle approach proposed by Biham, Biryukov, and Shamir (1999b); that is, the attacker seeks for two differentials (characteristics) with probability of one such that the last round of one differential coincides with the first round of the other and the intermediate differences at this round contradict. More precisely, suppose that an i-round differential $S = \Delta\alpha \xrightarrow{i} \Delta\gamma$ and a j-round

differential $T = \Delta\delta \xrightarrow{j} \Delta\beta$ represent two differentials (characteristics) over i and j rounds of a block cipher respectively. Further, assume that S and T satisfy the following:

- An input difference of $\Delta\alpha$ will terminate with a difference of $\Delta\gamma$ after i rounds of cipher with probability of one, i.e., with absolute certainty.
- An output difference of $\Delta\beta$ will terminate with a difference of $\Delta\delta$ after j rounds of decryption with probability of one.
- There exists at least a bit in which its intermediate difference $\Delta\gamma$ contradicts with that of $\Delta\delta$, that is, $\Delta\gamma \neq \Delta\delta$.

Under the above assumptions, it is apparent that a $(i+j)$-round differential (characteristic) commencing with $\Delta\alpha$ and terminating with $\Delta\beta$ has a zero chance of occurrence, that is, $\Pr(\Delta\alpha \xrightarrow{i+j} \Delta\beta) = 0$. Throughout this thesis, we denote such an $(i+j)$-round impossible differential (characteristic) as $\Delta\alpha \overset{i+j}{\not\rightarrow} \Delta\beta$. Figure 2.4 shows the construction of an impossible differential using the miss-in-the-middle approach.

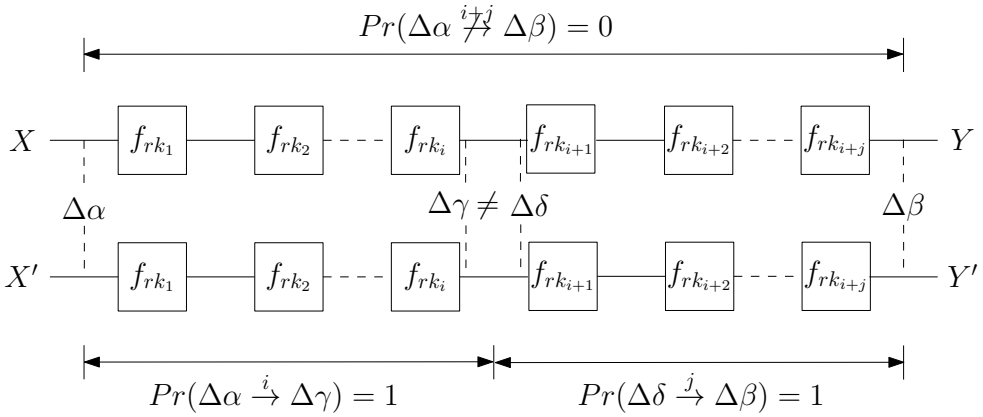


Figure 2.4: Impossible Differential: Miss-In-The-Middle Approach

Let $\mathtt{E} : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ be a cascade of three sub-ciphers, that is, $\mathtt{E} = \mathtt{E}_a \circ \mathtt{E}_0 \circ \mathtt{E}_b$ where $\mathtt{E}_0$ denotes the rounds for which $\Delta\alpha \not\rightarrow \Delta\beta$ holds, $\mathtt{E}_a$ and $\mathtt{E}_b$ denote the rounds before and after $\mathtt{E}_0$ respectively. For each guess

for the subkeys used in E_a and E_b, if an input difference produces a difference of α after E_a and its output difference produces a difference of β before E_b, then this key is discarded as the right subkey will not lead to such a differential. Thus, given a sufficient number of matching plaintext-ciphertext pairs, an attacker can discard all the wrong subkeys involved in the number of rounds before and after E_0.

3. Boomerang attack: The boomerang attack was first proposed by Wagner (1999) to show how differential-like attacks can be applied to break a block cipher even when differentials with either high or low probabilities do not exist. Instead of using a single differential on the entire cipher for differential cryptanalysis, boomerang attack utilises two differentials on two different parts of the cipher. As a result, the boomerang attack considers four pairs of plaintext and ciphertext, i.e., (P,C),(P',C'),(P^*,C^*) and (P'^*,C'^*). Let $\mathrm{E} : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ be a cascade of two sub-ciphers, that is, $\mathrm{E}_K = \mathrm{E}_K^0 \circ \mathrm{E}_K^1$ where E_K^0 represents the first half of the cipher and E_K^1 represents the last half of the cipher.

   Consider the two differentials $\Delta\alpha \to \Delta\beta$ with probability p for E_K^0 and $\Delta\delta \to \Delta\gamma$ with probability q for the inverse of E_K^1, that is, $(\mathrm{E}_K^1)^{-1}$. If the pair (P,P^*) fulfills the differential $\Delta\alpha \to \Delta\beta$ while the pairs (P,P') and (P^*,P'^*) fulfill the differential $\Delta\delta \to \Delta\gamma$, then the pair $(E_K^0(P'),E_K^0(P'^*))$ will satisfy the differential $\Delta\beta \to \Delta\alpha$ with probability $p^2 \cdot q^2$ for the inverse of E_K^0, that is, $(\mathrm{E}_K^0)^{-1}$ (as shown in Figure 2.5(a)) since

$$\begin{aligned}
\mathrm{E}_K^0(P') \oplus \mathrm{E}_K^0(P'^*) &= \mathrm{E}_K^0(P) \oplus \mathrm{E}_K^0(P^*) \oplus \mathrm{E}_K^0(P) \oplus \mathrm{E}_K^0(P') \oplus \mathrm{E}_K^0(P^*) \oplus \\
&\quad \mathrm{E}_K^0(P'^*) \\
&= \mathrm{E}_K^0(P) \oplus \mathrm{E}_K^0(P^*) \oplus (\mathrm{E}_K^1)^{-1}(C) \oplus (\mathrm{E}_K^1)^{-1}(C') \oplus \\
&\quad (\mathrm{E}_K^1)^{-1}(C^*) \oplus (\mathrm{E}_K^1)^{-1}(C'^*) \\
&= \Delta\beta \oplus \Delta\gamma \oplus \Delta\gamma = \Delta\beta. \qquad (2.1)
\end{aligned}$$

   Observe that if one sets the three differentials properly, then the last differential will be fulfilled accordingly. This is the reason why Wagner named his

attack as the boomerang attack: when you send it properly, it always comes back to you. Such a correct quartet is usually called a boomerang distinguisher.

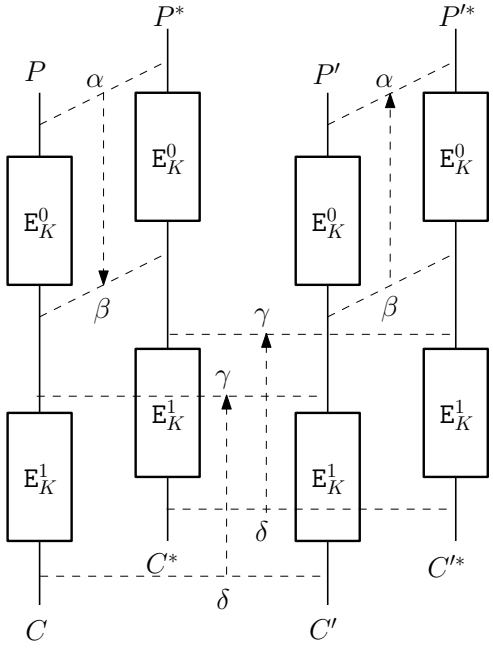
To launch a boomerang attack, the attacker first collects N pairs of plaintexts (P, P^*) where $P^* = P \oplus \Delta\alpha$ and obtain the corresponding ciphertexts (C, C^*) under a same secret key K. Subsequently, the attacker computes $C' = C \oplus \Delta\delta$ and $C'^* = C^* \oplus \Delta\delta$. Lastly, the attacker requests for the plaintexts (P', P'^*). It follows that $N \cdot p^2 q^2$ right quartets can be obtained.

However, for a randomly chosen function (instead of a block cipher E), the expected number of right quartets that fulfills $P' \oplus P'^* = \Delta\alpha$ is approximately $N \cdot 2^{-n}$. Thus, if $p \cdot q > 2^{-\frac{n}{2}}$, the boomerang distinguisher can efficiently distinguish between a block cipher E and a randomly chosen function given a sufficient number of adaptive chosen plaintexts and ciphertexts.

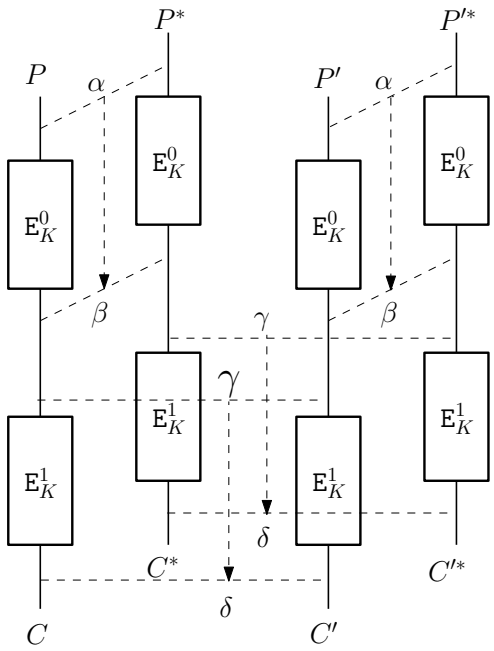
4. Amplified boomerang attack: To relax the requirement of adaptive chosen plaintext and ciphertext queries in a boomerang attack, Kelsey et al. (2001) introduced a variant of boomerang attack called an amplified boomerang attack. By making use of an amplified boomerang distinguisher shown in Figure 2.5(b), an amplified boomerang attack only requires chosen plaintext queries.

   Just as the boomerang distinguisher, let $\mathrm{E} : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ be a cascade of two sub-ciphers, that is, $\mathrm{E}_K = \mathrm{E}^0_K \circ \mathrm{E}^1_K$ where E^0_K represents the first half of the cipher and E^1_K represents the last half of the cipher. Consider the two differentials $\Delta\alpha \to \Delta\beta$ with probability p for E^0_K and $\Delta\gamma \to \Delta\delta$ with probability q for E^1_K. If the pairs (P, P^*) and (P', P'^*) fulfill the differential $\Delta\alpha \to \Delta\beta$, then the pairs $(\mathrm{E}^0_K(P), \mathrm{E}^0_K(P'))$ and $(\mathrm{E}^0_K(P^*), \mathrm{E}^0_K(P'^*))$will fulfill the differential $\Delta\gamma \to \Delta\delta$ with probability $2^{-n} \cdot p^2 \cdot q^2$ for E^1_K since $\mathrm{E}_K(P) \oplus \mathrm{E}_K(P') = \Delta\gamma$ with probability 2^{-n} and

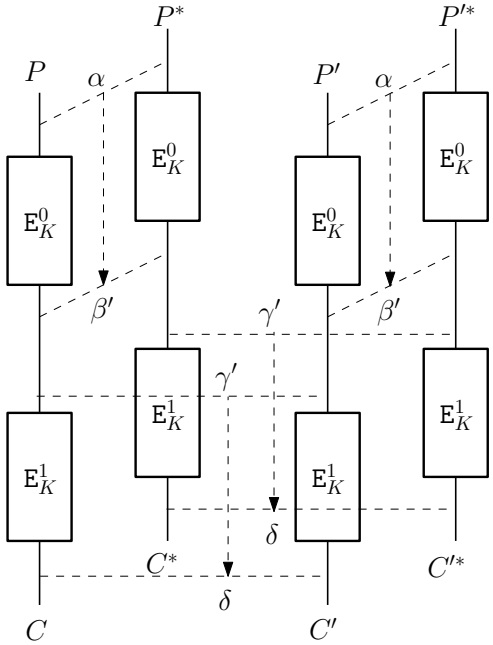
$$\begin{aligned} \mathrm{E}^0_K(P^*) \oplus \mathrm{E}^0_K(P'^*) &= \mathrm{E}^0_K(P) \oplus \mathrm{E}^0_K(P^*) \oplus \mathrm{E}^0_K(P') \oplus \mathrm{E}^0_K(P'^*) \oplus \mathrm{E}^0_K(P) \oplus \\ &\quad \mathrm{E}^0_K(P') \\ &= \Delta\beta \oplus \Delta\beta \oplus \Delta\gamma = \Delta\gamma. \end{aligned} \tag{2.2}$$



a) Boomerang



b) Amplified Boomerang



C) Rectangle

$\beta' =$ any possible value of β,
$\delta' =$ any possible value of δ,
$\beta' \neq \delta'$.

Figure 2.5: The Boomerang, Amplified Boomerang and Rectangle Distinguishers

However, for a randomly chosen function (instead of a block cipher E), a right quartet that satisfies $\mathrm{E}_K(P) \oplus \mathrm{E}_K(P') = \mathrm{E}_K(P^*) \oplus \mathrm{E}_K(P'^*) = \Delta\gamma$ is approximately 2^{-2n}. Thus, if $p \cdot q > 2^{-\frac{n}{2}}$, the amplified boomerang distinguisher can efficiently distinguish between a block cipher E and a randomly chosen function given a sufficient number of chosen plaintexts.

5. Rectangle attack: In order to reduce the data complexity needed in an amplified boomerang attack, Biham et al. (2001) improved the amplified boomerang attack and named their variant as a rectangle attack. Specifically, the rectangle distinguisher allows $\Delta\beta$ to take any possible value of $\Delta\beta'$ and $\Delta\delta$ to take any possible value of $\Delta\delta'$ as long as $\Delta\beta' \neq \Delta\delta'$ as shown in Figure 2.5(c). Thus, a right quartet will be formed with the probability of $2^{-n} \cdot \hat{p}^2 \cdot \hat{q}^2$ where $\hat{p} = (\sum_{\Delta\beta'} \mathrm{Pr}^2_{\mathrm{E}^0_K}(\Delta\alpha \rightarrow \Delta\beta'))^{\frac{1}{2}}$ and $\hat{q} = (\sum_{\Delta\gamma'} \mathrm{Pr}^2_{\mathrm{E}^1_K}(\Delta\gamma' \rightarrow \Delta\delta))^{\frac{1}{2}}$. Note that $\hat{p} \geq p$ and $\hat{q} \geq q$. Thus, a smaller amount of data is needed to form a right quartet.

6. Related-key attacks: Related-key cryptanalysis was independently introduced by Knudsen (1993) and Biham (1994). The principle of related-key cryptanalysis is to study how a particular input difference can affect a particular output difference where the outputs are obtained by encrypting the inputs under two different unknown secret keys with a known particular difference. Similar to differential cryptanalysis, the difference can be an exclusive-or operation or a modular addition difference.

   A related-key differential characteristic is a path that specifies the propagation from a particular input difference to a particular output difference under two different unknown secret keys. Once again, it is likely that more than one related-key differential characteristic with the same input and output differences but different intermediate differences exist. Joining all related-key differential characteristics forms a related-key differential. As stated by Lu (2008), the probability of an i-round related-key differential for a block cipher under two different unknown secret keys (K, K') is defined in a similar way as a differential characteristic under a fixed unknown secret key.

On the other hand, a related-key amplified boomerang attack was introduced by Kim, Kim, Hong, Lee, and Hong (2004) by combining the techniques of amplified boomerang and related-key attacks. Such related-key amplified boomerang attacks (Biham, Dunkelman, & Keller, 2005b; S. Hong, Kim, Lee, & Preneel, 2005; Kim, Hong, & Preneel, 2007; Kim, Kim, Lee, Lim, & Song, 2005; Lu, 2009; Lu & Kim, 2008; Lu, Kim, Keller, & Dunkelman, 2006; Lu, Lee, & Kim, 2006) were then used to attack AES, KASUMI, XTEA, SHACAL-1 and SHACAL-2.In addition, Biham, Dunkelman, and Keller (2005a) independently introduced the related-key rectangle attack and both results (Kim et al., 2004; Biham et al., 2005a) had been combined to result in a more detailed related-key rectangle attack (Kim et al., 2012).

Let $\mathrm{E} : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ be a cascade of two sub-ciphers, that is, $\mathrm{E}_K = \mathrm{E}^0_K \circ \mathrm{E}^1_K$ where E^0_K represents the first half of the cipher and E^1_K represents the last half of the cipher. A typical related-key rectangle distinguisher involves four related-keys K_A, K_B, K_C and K_D which satisfy $K_A \oplus K_B = K_C \oplus K_D = \Delta K_0$ and $K_A \oplus K_C = K_B \oplus K_D = \Delta K_1$ where ΔK_0 and ΔK_1 are two known differences. Instead of using two differentials in a rectangle attack, a related-key rectangle attack uses four related-key differentials shown in Figure 2.6 as follows:

- All the possible related-key differentials $(\Delta\alpha \rightarrow \Delta\beta)$ for E^0 are under the related-keys K_A and K_B
- All the possible related-key differentials $(\Delta\alpha \rightarrow \Delta\beta)$ for E^0 are under the related-keys K_C and K_D
- All the possible related-key differentials $(\Delta\gamma \rightarrow \Delta\delta)$ for E^1 are under the related-keys K_A and K_B
- All the possible related-key differentials $(\Delta\gamma \rightarrow \Delta\delta)$ for E^1 are under the related-keys K_C and K_D

If the pairs $(P, P^* = P \oplus \Delta\alpha)$ and $(P', P'^* = P' \oplus \alpha)$ fulfill the following differentials:

- $\mathrm{E}^0_{K_A}(P) \oplus \mathrm{E}^0_{K_B}(P^*) = \mathrm{E}^0_{K_C}(P') \oplus \mathrm{E}^0_{K_D}(P'^*) = \Delta\beta,$

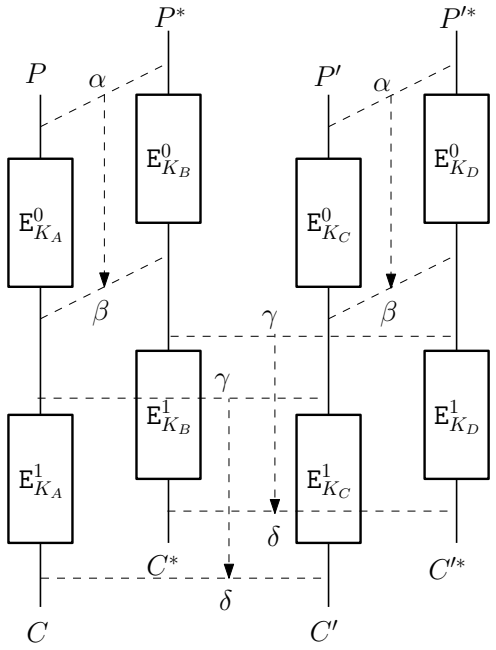


Figure 2.6: The Related-Key Amplified Boomerang Distinguisher

- $\mathrm{E}^0_{K_A}(P) \oplus \mathrm{E}^0_{K_C}(P') = \Delta\gamma$,
- $\mathrm{E}_{K_A}(P) \oplus \mathrm{E}_{K_C}(P') = \mathrm{E}_{K_B}(P^*) \oplus \mathrm{E}_{K_D}(P'^*) = \Delta\delta$,

where $\Delta\beta \neq \Delta\gamma$, then the pairs $(\mathrm{E}^0_{K_B}(P^*), \mathrm{E}^0_{K_D}(P'^*))$will fulfill the differential $(\Delta\gamma \rightarrow \Delta\delta)$ with probability $2^{-n} \cdot p^2 \cdot q^2$ for E^1_K since $\mathrm{E}^0_{K_A}(P) \oplus \mathrm{E}^0_{K_C}(P') = \Delta\gamma$ with probability 2^{-n} and

$$\begin{aligned} \mathrm{E}^0_{K_B}(P^*) \oplus \mathrm{E}^0_{K_D}(P'^*) &= \mathrm{E}^0_{K_A}(P) \oplus \mathrm{E}^0_{K_B}(P^*) \oplus \mathrm{E}^0_{K_C}(P') \oplus \mathrm{E}^0_{K_D}(P'^*) \oplus \mathrm{E}^0_{K_A}(P) \oplus \\ &\quad \mathrm{E}^0_{K_c}(P') \\ &= \Delta\beta \oplus \Delta\beta \oplus \Delta\gamma = \Delta\gamma. \end{aligned} \tag{2.3}$$

Note that $p = \Pr_{\mathrm{E}^0_{K_A}, \mathrm{E}^0_{K_B}}(\Delta\alpha \rightarrow \Delta\beta) = \Pr_{\mathrm{E}^0_{K_C}, \mathrm{E}^0_{K_D}}(\Delta\alpha \rightarrow \Delta\beta)$ and $q = \Pr_{\mathrm{E}^1_{K_A}, \mathrm{E}^1_{K_C}}(\Delta\gamma \rightarrow \Delta\delta) = \Pr_{\mathrm{E}^1_{K_B}, \mathrm{E}^1_{K_D}}(\Delta\gamma \rightarrow \Delta\delta)$.

However, for a randomly chosen function (instead of a block cipher E), a right quartet that fulfills the above related-key differentials is approximately 2^{-2n}. Thus, if $p \cdot q > 2^{-\frac{n}{2}}$, the related-key rectangle distinguisher can efficiently distinguish between a block cipher E and a randomly chosen function given a

sufficient number of chosen plaintext queries. If $\Delta K_0 = 0$ or $\Delta K_1 = 0$, then the related-key rectangle distinguisher will involve two related-keys only.

Note that in addition to those assumptions (Lai, Massey, & Murphy, 1991) used in differential cryptanalysis (Biham & Shamir, 1991b), the related-key amplified boomerang attack requires another assumption about independence, and we refer the reader to (Murphy, 2011; Kim et al., 2012) for a more formal discussion of the assumptions as well as the attack technique. These assumptions result in cases where the probability of a related-key amplified boomerang distinguisher may be overestimated or underestimated, and similarly for the success probability of the attack. Nonetheless, it seems reasonable to consider the worst case assumption from the user point of view. An application of such an attack was proposed by Dunkelman, Keller, and Shamir (2010) to break the full KASUMI cipher with a practical complexity, and its validity was experimentally verified.

7. Weak keys attack: According to Handschuh and Preneel (2008), a key is known as a weak key if the following two conditions are met:

   - Under this key, the algorithm works in an unexpected manner.
   - It is relatively easy for the attacker to test for the existence of this key.

   A class of weak keys is then called a weak key class if all the keys in this class result in a similar behaviour. In addition, it should typically require much lower time complexities as compared to the number of keys in the class for an attacker to test if a particular key lies in the class. The use of a block cipher is limited if the block cipher possesses a high ratio of weak keys especially when the cipher is used in a setting which requires the change of keys frequently.

2.2 Applications of Block Cipher

2.2.1 Message Authentication Code

Message authentication codes (MACs) can be viewed as the symmetric key equivalent of digital signatures. Essentially, MAC is a symmetric key cryptographic

scheme that serves to authenticate both the source of a message and its integrity. It is widely used in the mobile, wireless and data communication networks due to its high efficiency.

At present, there exist several approaches to design a MAC scheme. In particular, MACs can be constructed based on cryptographic hash functions (e.g., HMAC (Bellare, Canetti, & Krawczyk, 1996)), block ciphers (e.g., CBC-MAC (ISO, 1999), XCBC (Black & Rogaway, 2000), TMAC (Kurosawa & Iwata, 2003), OMAC (Iwata & Kurosawa, 2003)) or even universal hash functions (e.g., UMAC (Black, Halevi, Krawczyk, Krovetz, & Rogaway, 1996) and MMH (Halevi & Krawczyk, 1997)). Among these MAC schemes, CBC-MAC and HMAC are the most popular. However, these two schemes share a common characteristic of being inherently sequential where one can only process the i-th message block after all the previous message blocks have been processed.

A MAC algorithm processes a message M with an arbitrary length to generate an τ-bit tag T_M under the control of a k-bit secret key K. To be precise, a MAC algorithm (Handschuh & Preneel, 2008) consists of the following three algorithms:

1. A key generation algorithm, `KeyGen`: A randomised algorithm that generates a k-bit secret key $K \in \mathscr{K}$.
2. A tag generation algorithm, `Tag`: An algorithm which takes the inputs a message $M \in \mathscr{M}$ and a secret key K to generate a τ-bit tag $T_M \in \mathscr{T}$.
3. A tag verification algorithm, `Verify`: An algorithm which takes the inputs a message M, a tag T_M and a secret key K to generate an answer true or false (1/0).

We consider the standard model for the security of a MAC in the presence of a chosen message attack, in which an attacker is given access to the tag generation oracle and tag verification oracle. The attacker can query any tag for any messages he chooses using the tag generation oracle. At the same time, he can submit a tag-message pair to

the tag verification oracle for verification purpose as well. Such queries are termed as tagging queries and verification queries respectively. The attacker is assumed to win the game if he can forge a tag T_M for a new message M that had not been queried by the tag generation oracle. Such a pair (M, T_M) is called a forgery.

In the standard model of evaluating MAC security, there are two main types of attacks on MACs: key recovery attack and forgery attack (Jia, Wang, Yuan, & Xu, 2009). For the key recovery attack, an attacker tries to recover the secret key K from a number of tags. Obviously, a successful key recovery attack leads to the construction of an arbitrary number of forgeries. Ideally, any attack following key recovery requires about 2^k operations. On the other hand, a forgery attack (without the knowledge of K) can be divided into the following three types:

1. Existential forgery: An attacker is able to get a corresponding tag for a message M which has not been generated by the user. Here, the message M may not have any particular meaning.

2. Selective forgery: An attacker is able to determine the tag for a message of his choice prior to the attack without querying this message with the tag generation oracle.

3. Universal forgery: An attacker is able to find a tag for any message without querying it with the tag generation oracle. It is obvious that this attack is more powerful than the above two attacks.

In this thesis, we are primarily concerned with the existential forgery attack and this leads us to the following notion of security called existential unforgeability under chosen message attack against MACs (Dodis, Kiltz, Pietrzak, & Wichs, 2012). We briefly describe this notion and readers can refer to Dodis et al. (2012) for more details. However, we ignore the verification oracle (chosen verification query) in the following model since our attacks do not involve any verification query.

Existential unforgeability: Consider a MAC: $\mathscr{K} \times \mathscr{M} \rightarrow \mathscr{T}$. The attacker can make queries to the oracle, namely $\texttt{Tag}_K(\cdot)$. $\texttt{Tag}_K(\cdot)$ is an oracle that allows the attacker to obtain tags of messages of his choice. We name $\texttt{Tag}_K(\cdot)$ the tagging oracle and queries to $\texttt{Tag}_K(\cdot)$ the tagging queries.

The query complexity includes the total number of queries made by the attacker, q, the total number of message blocks in all q queries, $\sigma = \sum_{i=1}^{q} l_i$ (l_i denotes the number of message blocks of i-th query), and the maximum number of message blocks of the longest query made by the attacker, $l = \max l_i$ for $1 \leq i \leq q$.

The security game proceeds as follows:

1. The challenger runs KeyGen to select a secret key K randomly.
2. Query Phase: The attacker $\mathscr{A}$ is given access to the oracle $\texttt{Tag}_K(\cdot)$. For $1 \leq i \leq q$, the challenger returns $\texttt{Tag}_K(M_i) = T_{M_i}$ for a tagging query. The queries are adaptive where $\mathscr{A}$ can view the response of the previous queries before sending the next query.
3. Forgery Attempt: The attacker $\mathscr{A}$ generates a forgery $T_{M'}$ on M'.

An attacker wins the game if

- $\texttt{Verify}_K(M', T_{M'}) = 1$ and
- the attacker had not queried $\texttt{Tag}_K(\cdot)$ with the message M'.

Remark (Jia et al., 2009): An attack is considered as an almost universal forgery attack when an attacker is able to forge the tag for *almost* all the messages.

2.2.2 Authenticated Encryption/Mode of Operation

An authenticated encryption scheme (Bellare & Namprempre, 2008; Black, 2005) is a symmetric encryption scheme to provide both confidentiality and authenticity. An authenticated encryption scheme consists of the following three algorithms:

1. A key generation algorithm, `KeyGen`: A randomised algorithm that generates a k-bit secret key K.
2. An encryption algorithm, `Enc`: An algorithm which takes the inputs a plaintext P, an initial vector IV (also known as nonce), an optional authenticated data A and a secret key K to generate a ciphertext C and a τ-bit tag T.
3. A decryption algorithm, `Dec`: An algorithm which takes the inputs a ciphertext C, an initial vector IV, an optional authenticated data A, a tag T and a secret key K to return the plaintext P or the specific symbol **FAIL**.

In the standard model of evaluating authenticated encryption security (Bellare & Namprempre, 2000, 2008; Rogaway, 2002), we consider the security notion for authenticated encryption regarding authenticity only (excludes the notion of privacy as we only deal with the security of authenticated encryption on authenticity in this thesis). For authenticity, the attacker $\mathscr{A}$ is given access to both encryption and decryption oracles. The encryption oracle takes (P, IV, A) as inputs and returns (C, T). On the other hand, the decryption oracle takes (C, IV, A, T) as inputs and returns P or **FAIL**. $\mathscr{A}$ is free to issue any query to these two oracles and the query complexity includes the number of queries, the total length of queries and the maximum length of queries.

There are differences between the standard models proposed by Bellare and Namprempre (2000, 2008), Rogaway (2002) and McGrew and Viega (2005). McGrew and Viega (2005) assumed that $\mathscr{A}$ is nonce-respecting where $\mathscr{A}$ does not make two queries with the same nonce/IV to the same oracle (though he is free to submit a value to both oracles). However, it is usually acceptable that $\mathscr{A}$ is forbidden to reuse nonce

in the encryption oracle, but not the decryption oracle. This is the notion proposed by Bellare and Namprempre (2000, 2008) and Rogaway (2002). For ease of understanding, the assumptions made in each of the attacks presented in this thesis will be listed down in the related chapters only.

2.2.3 Image Encryption

Due to the fast development of internet and network technology, multimedia data (i.e., image and video) has been transmitted between users more frequently and in vast proportions. In certain applications, images sent over the internet or any communication channel must be protected from being captured by any unauthorized users. For example, cable TV providers, online personal photograph albums, medical images and sensitive military images. Thus, image encryption is a crucial issue in modern times as information exchange continues to proliferate.

Image encryption was introduced to encrypt a plain image P to an encrypted image C under the control of a secret key K. Without a proper secret key, any unauthorised users are not able to recover the plain image even if he can obtain the encrypted image by any means. General cryptographic primitives play an important role in providing confidentiality to images, such as block ciphers, stream ciphers and public key cryptographic schemes. However, different methods had been proposed to encrypt images in the literature. For instance, due to the statistical properties of chaotic systems, chaos theory had been used extensively in proposing image encryption schemes.

Due to the strong correlation among pixels in a plain image, block ciphers (under electronic codebook mode) are not suitable for image encryption. Conceptually, an image encryption algorithm is very similar to a block cipher as a weak round function f is applied for a number of rounds to encrypt a plain image P into an encrypted image C using the `E` algorithm under subkeys that are derived from a secret key using the `Key Schedule` algorithm. Even though most of the image encryption algorithms do not specifically describe their `Key Schedule` algorithms explicitly as compared to block ciphers, an image encryption algorithm can similarly be seen to comprise of

three different algorithms, i.e., `Key Schedule`, `E` and `D`. However, even though an image encryption provides confidentiality, the image encryption schemes are not analysed based on constructive cryptographic approaches. In this thesis, we analyse the security of image encryption schemes from cryptographic perspectives, i.e., security requirements of a block cipher.

2.3 Notation

Throughout this thesis, the elements of the finite field $GF(2^n)$ will be represented either by their polynomial representation or by the corresponding n-bit binary string of coefficients. Here, we fix a basis $1, x, \ldots, x^{n-1}$ for the vector space $GF(2^n)$ over $GF(2)$. A difference with prefix $0x$ is in hexadecimal (base 16) notation and a difference without prefix $0x$ is in binary (base 2) notation. In this thesis, the following notations are adopted.

- $\oplus$: Bitwise logical exclusive-or (XOR) of two bit strings of the same length.
- & or $\cap$: Bitwise logical AND of two bit strings of the same length.
- $\cup$: Bitwise logical AND of two bit strings of the same length.
- $\boxplus$: Addition modulo 2^{32}.
- $\boxminus$: Subtraction modulo 2^{32}.
- $\cdot$: Multiplication in a finite field.
- $||$: Bit string concatenation.
- $\lll$: Left rotation of a bit string.
- $\ggg$: Right rotation of a bit string.
- $\lfloor x \rfloor$: The largest integer that is less than or equal to x.
- $\lceil x \rceil$: The smallest integer that is greater than or equal to x.
- $|x|$: The bitlength of bit string x.

- $|x|_8$: The byte length of bit string x.
- 0^i: The bit string that consists of i '0' bits.
- $\circ$: Functional composition. When composing functions X and Y, $Y \circ X$ denotes the function obtained by first applying X and then applying Y.
- e: The base of the natural logarithm ($\mathsf{e} = 2.71828\cdots$).
- $\star$: An arbitrary value of some length, where two values represented by the $\star$ symbol may be different.
- $\mathtt{E}_K(x)$: The encryption of bit string x using a block cipher $\mathtt{E}$ under the control of a secret key K.
- $\mathrm{MSB}_i(x)$: The bit string consisting of the i left-most bits of the bit string x.
- $[x]_i$: The binary representation of the nonnegative integer x as a string of i bits, where $x < 2^i$.
- $\mathrm{pad}(i)$: Given a bit string i and $|i| < n$, the padding of i is the bit string $i10^{n-|i|-1}$. If $|i| = n$, $\mathrm{pad}(i) = i$. Note that n is the block size of a block cipher $\mathtt{E}$.
- $\mathrm{ntz}(i)$: The number of trailing 0-bits in the bit string i.
- τ: The tag length is an integer $\tau \in [1, n]$ where n is the block size of a block cipher $\mathtt{E}$.

For ease of reading and understanding, we may repeat the definition of the notation throughout the thesis.

CHAPTER 3

WEAK KEYS OF THE FULL MISTY1 BLOCK CIPHER FOR RELATED-KEY CRYPTANALYSIS

The MISTY1 block cipher has a 64-bit block length, a 128-bit secret key and a recommended number of 8 rounds. It is a Japanese CRYPTREC-recommended e-government cipher, a European NESSIE selected cipher and an ISO standard. Despite considerable cryptanalytic efforts, there has been no published cryptanalytic attack on the full MISTY1 cipher algorithm until 2013. In this chapter, we present related-key differential and related-key amplified boomerang attacks on the full MISTY1 under certain weak key assumptions. Specifically, we present the following attacks:

- $2^{103.57}$ weak keys and a related-key differential attack on the full MISTY1 with a data complexity of 2^{61} chosen ciphertexts and a time complexity of $2^{90.93}$ encryptions, and
- 2^{92} weak keys and a related-key amplified boomerang attack on the full MISTY1 with a data complexity of $2^{60.5}$ chosen plaintexts and a time complexity of $2^{87.33}$ encryptions.

Our results are the first to exhibit a cryptographic weakness in the full MISTY1 cipher (when used with the recommended 8 rounds) even though that Todo (2015) found the first integral attack on full MISTY1 under single secret key scenario. More importantly, we show that the MISTY1 cipher is distinguishable from an ideal cipher and thus cannot be regarded as an ideal cipher.

3.1 Introduction

The MISTY1 block cipher (Matsui, 1997) has a 64-bit block length, a 128-bit secret key and a variable number of rounds; the officially recommended number of rounds is 8. We consider the version of MISTY1 that uses the recommended 8 rounds in this chapter, which is also the most widely discussed version so far. MISTY1 has a Feistel structure with a total of ten key-dependent logical functions **FL** — two **FL** functions at the beginning plus two inserted after every two rounds. It became a CRYPTREC e-government recommended cipher (CRYPTREC, 2003) and a NESSIE selected block cipher (NESSIE, 2004). In addition, MISTY1 was adopted as an ISO standard (ISO, 2005, 2010).

MISTY1 has attracted extensive attention since its publication and its security has been analysed against a wide range of cryptanalytic techniques (Babbage & Frisch, 2000; Chen & Dai, 2011; Dai, 2012; Dai & Chen, 2012; Kühn, 2001, 2002; E. Lee et al., 2008; Lu, Kim, Keller, & Dunkelman, 2008; Sun & Lai, 2009; Tanaka, Hatano, Sugio, & Kaneko, 2007; Tsunoo, Saito, Nakashima, & Shigeri, 2009a, 2009b, 2010, 2012). In summary, the main cryptanalytic results on MISTY1 published to date are as follows. Dunkelman and Keller (2008) described impossible differential attacks (Knudsen, 1998; Biham et al., 1999a) on 6-round MISTY1 with **FL** functions and 7-round MISTY1 without **FL** functions. Subsequently, E. Lee et al. (2008) gave a related-key amplified boomerang attack (Kim et al., 2004; Biham et al., 2005a; S. Hong et al., 2005) on 7-round MISTY1 with **FL** functions under a class of 2^{73} weak keys[1] and Tsunoo et al. (2009b) presented a higher-order differential attack (Lai, 1994; Knudsen, 1995) on 6 and 7-round MISTY1 with **FL** functions (without making a weak key assumption). Then, Sun and Lai (2009) presented an integral attack on 6-round MISTY1 with **FL** functions, building on Knudsen and Wagner's integral attack (Knudsen & Wagner, 2002) on 5-round MISTY1. Following the work done by E. Lee et al., Chen and Dai (2011) presented a 7-round related-key amplified boomerang distinguisher with probability 2^{-118} under a class of 2^{90} weak keys

[1] A class of weak keys is defined as a class of keys under which the concerned cipher is more vulnerable to be attacked.

and gave a related-key amplified boomerang attack on the 8-round MISTY1 with only the first 8 **FL** functions. Furthermore, they described a 7-round related-key differential with probability 2^{-58} under a class of $2^{102.57}$ weak keys and finally presented a related-key differential attack on the 8-round MISTY1 with only the last 8 **FL** functions (Dai & Chen, 2012). Later, Jia and Li (2012) improved the results obtained by Dunkelman and Keller (2008) and successfully extended impossible differential attack on 7-round MISTY1 without first two **FL** functions. So far, there does not exist any published (non-generic) cryptanalytic attack on the full 8 rounds of MISTY1 until 2012. In CRYPTO 2015, Todo (2015) constructed a new integral characteristic by using the propagation characteristic of the division property and successfully recovered the secret key of the full MISTY1 by exploiting the new integral characteristic. The proposed integral attack on full MISTY1 is a theoretical attack since almost all plaintexts are needed to launch such integral attack.

Related-key cryptanalysis (Biham, 1994; Knudsen, 1993) assumes that the attacker knows the relationship between one or more pairs of unknown keys; certain current real-world applications may allow for practical related-key attacks, for example, key-exchange protocols (Kelsey et al., 1996). Related-key differential cryptanalysis (Kelsey et al., 1996) takes advantage of how a particular input difference can affect a particular output difference where the outputs are obtained by encrypting the inputs under two different secret keys with a known particular difference. The related-key amplified boomerang attack (Biham et al., 2005a; S. Hong et al., 2005; Kim et al., 2004) is a combination of related-key cryptanalysis and the amplified boomerang attack (Kelsey et al., 2001). The amplified boomerang attack is a variant of the boomerang attack (Wagner, 1999) that is based on differential cryptanalysis (Biham & Shamir, 1991a). Remarkably, the related-key differential cryptanalysis technique was used by Biryukov, Khovratovich, and Nikolić (2009) to yield the first cryptanalytic attack on the full version of the AES block cipher (NIST, 2001) with 256 key bits under certain weak key assumptions. Furthermore, the related-key amplified boomerang attack technique was used to yield the first cryptanalytic attacks on the full versions of both AES with 192 and 256 key bits and KASUMI (3GPP,

2001) which is a variant of MISTY1, without making any weak key assumption, by Biham et al. (2005b), Biryukov and Khovratovich (2009) and Dunkelman et al. (2010) respectively.

In this chapter, we show for the very first time that the full MISTY1 cipher can be distinguished from an ideal cipher (in the related-key model) mainly from a theoretical perspective[2]. Building on the work done by Chen and Dai (2011) and Dai and Chen (2012), we present related-key differential and amplified boomerang attacks on the full MISTY1 cipher under certain weak key assumptions. First, we show that Dai and Chen's 7-round related-key differential can be used to break the full MISTY1 under the class of $2^{102.57}$ weak keys and observe that there exists also a different class of $2^{102.57}$ weak keys under which similar results hold. Finally, we find that under the class of 2^{90} weak keys described by Chen and Dai (2011), Chen and Dai's 7-round related-key amplified boomerang distinguisher actually has a probability of 2^{-116}, instead of 2^{-118}, which can be used to attack the full MISTY1; and similar results hold for three other classes of weak keys of the same size. Table 3.1 summarises our results as well as the previously published main cryptanalytic results on MISTY1, where CP and CC refer respectively to the numbers of chosen plaintexts and chosen ciphertexts while Enc. refers to the required number of encryption operations of the relevant version of MISTY1.

Organisation. The remainder of the chapter is organised as follows. In the next section, we describe the notation and the MISTY1 cipher. In Sections 3.3 and 3.4, we review Chen and Dai's cryptanalytic results and give our differential and amplified boomerang cryptanalytic results on MISTY1, respectively. Section 3.5 concludes this chapter.

[2]Our results on full MISTY1 were dated in 2013 and no attacks had been found on full MISTY1 under single secret key scenario until 2015 by Todo

Table 3.1: Main Cryptanalytic Results on MISTY1 with FL Functions

#Rounds	#Keys	Attack Type	Data	Memory	Time	Source
$6\ (1-6)$	2^{128}	Impossible differential	2^{51}CP	not specified	$2^{123.4}$Enc.	(Dunkelman & Keller, 2008)
$6\ (1-6)$	2^{128}	Higher-order differential	$2^{53.7}$CP	not specified	$2^{64.4}$Enc.	(Tsunoo et al., 2009a, 2009b)
$6\ (3-8)$	2^{128}	Integral	2^{32}CC	not specified	$2^{126.1}$Enc.	(Sun & Lai, 2009)
$7\ (1-7)$	2^{128}	Higher-order differential	$2^{54.1}$CP	not specified	$2^{120.7}$Enc.	(Tsunoo et al., 2009b, 2010)
$7^{\dagger}\ (1-7)$	2^{128}	Impossible differential	2^{58}CP	not specified	$2^{124.4}$Enc..	(Jia & Li, 2012)
$7^{\dagger}\ (2-8)$	2^{73}	Related-key amplified boomerang	2^{54}CP	2^{59}Bytes	$2^{55.3}$Enc.	(Lu et al., 2008)
$8^{\dagger}\ (1-8)$	2^{90}	Related-key amplified boomerang	2^{63}CP	2^{65}Bytes	2^{70}Enc.	(Chen & Dai, 2011)
$8^{\dagger}\ (1-8)$	$2^{102.57}$	Related-key differential	2^{61}CC	2^{34}Bytes	$2^{84.6}$Enc.	(Dai & Chen, 2012)
Full	2^{128}	Integral attack	$2^{63.58}$CP	not specified	2^{121}Enc.	(Todo, 2015)
Full	2^{128}	Integral attack	$2^{63.994}$CP	not specified	$2^{107.9}$Enc.	(Todo, 2015)
Full	$2^{103.57}$	Related-key differential	2^{61}CC	$2^{99.2}$Bytes	$2^{90.93}$Enc.	Section 3.3‡
	2^{92}	Related-key amplified boomerang	$2^{60.5}$CP	$2^{80.07}$Bytes	$2^{87.33}$Enc.	Section 3.4‡

†: Exclude the first or last two FL functions; ‡: Complexity is only for one class of weak keys.

3.2 The MISTY1 Block Cipher

MISTY1 (Matsui, 1997) employs a complex Feistel structure with a 64-bit block length and a 128-bit secret key. It uses the following three functions **FL**, **FI**, **FO**, which are respectively depicted in Figure 3.1-(a), Figure 3.1-(b) and Figure 3.1-(c) with their respective subkeys to be described below.

- **FL** : $\{0,1\}^{32} \times \{0,1\}^{32} \rightarrow \{0,1\}^{32}$ is a key-dependent linear function. If $X = (X_L||X_R)$ is a 32-bit block of two 16-bit words X_L, X_R, and $Y = (Y_1||Y_2)$ is a 32-bit block of two 16-bit words Y_1, Y_2, then output $\mathbf{FL}(X,Y) = (X_L \oplus ((X_R \oplus (X_L \cap Y_1)) \cup Y_2), X_R \oplus (X_L \cap Y_1))$.

- **FI** : $\{0,1\}^{16} \times \{0,1\}^{16} \rightarrow \{0,1\}^{16}$ is a non-linear function. If $X = (X_L||X_R)$ and $Y = (Y_1||Y_2)$ are 16-bit blocks, (here X_L, Y_2 are 9 bits long and X_R, Y_1 are 7 bits long), then $\mathbf{FI}(X,Y)$ is computed as follows, where $XL_0, XR_0, \cdots, XL_3, XR_3$ are 9 or 7-bit variables, S_9 is a 9×9-bit bijective S-box, S_7 is a 7×7-bit bijective S-box, the function Extnd extends from 7 bits to 9 bits by concatenating two zeros on the left side and the function Trunc truncates two bits from the left side.

  1) Set $XL_0 = X_L$ and $XR_0 = X_R$.
  2) Compute $XL_1 = XR_0$ and $XR_1 = S_9(XL_0) \oplus \text{Extnd}(XR_0)$.
  3) Compute $XL_2 = XR_1 \oplus Y_2$ and $XR_2 = S_7(XL_1) \oplus \text{Trun}(XR_1) \oplus Y_1$.
  4) Compute $XL_3 = XR_2$ and $XR_3 = S_9(XL_2) \oplus \text{Extnd}(XR_2)$.
  5) Output $\mathbf{FI}(X,Y) = (XL_3||XR_3)$.

- **FO** : $\{0,1\}^{32} \times \{0,1\}^{64} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$ is a non-linear function. If $X = (X_L||X_R)$ is a 32-bit block of two 16-bit words X_L, X_R, $Y = (Y_1||Y_2||Y_3||Y_4)$ is a 64-bit block of four 16-bit words Y_1, Y_2, Y_3, Y_4 and $Z = (Z_1||Z_2||Z_3)$ is a 48-bit block of three 16-bit words Z_1, Z_2, Z_3, then $\mathbf{FO}(X,Y,Z)$ is defined as follows, where $XL_0, XR_0, \cdots, XL_3, XR_3$ are 16-bit variables.

  1) Set $XL_0 = X_L$ and $XR_0 = X_R$.

2) For $j = 1,2,3$, compute $XL_j = XR_{j-1}$ and $XR_j = \mathbf{FI}(XL_{j-1} \oplus Y_j, Z_j) \oplus XR_{j-1}$.

3) Output $\mathbf{FO}(X,Y,Z) = (XL_3 \oplus Y_4)||XR_3$.

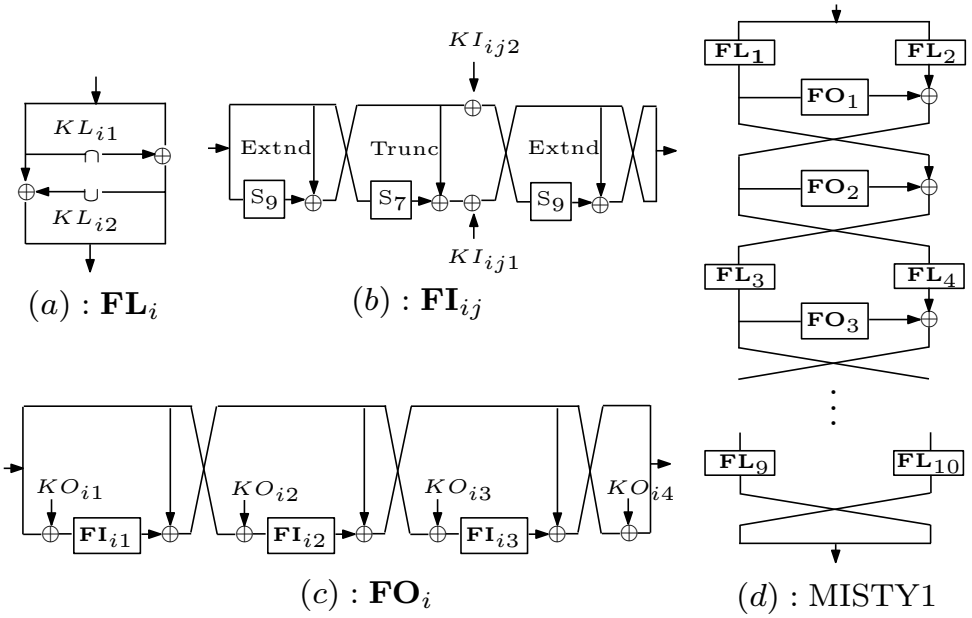


Figure 3.1: MISTY1 and Its Components

MISTY1 uses a total of ten 32-bit subkeys $KL_1, KL_2, \cdots, KL_{10}$ for the **FL** functions, twenty-four 16-bit subkeys KI_{ij} for the **FI** functions and thirty-two 16-bit subkeys KO_{il} for the **FO** functions, $(1 \leqslant i \leqslant 8, 1 \leqslant j \leqslant 3, 1 \leqslant l \leqslant 4)$, all derived from a 128-bit secret key K. The key schedule is as follows:

1) Represent K as eight 16-bit words $K = (K_1, K_2, \cdots, K_8)$.

2) Generate K' as eight 16-bit words $K' = (K'_1, K'_2, \cdots, K'_8)$ where $K'_i = \mathbf{FI}(K_i, K_{i+1})$, for $i = 1,2,\cdots,8$, where the subscript $i+1$ is reduced by 8 when it is larger than 8, (similarly for some subkeys in the following step).

3) Compute the subkeys as follows:

- $KO_{i1} = K_i, KO_{i2} = K_{i+2}, KO_{i3} = K_{i+7}, KO_{i4} = K_{i+4}$,

- $KI_{i1} = K'_{i+5}, KI_{i2} = K'_{i+1}, KI_{i3} = K'_{i+3}$,
- $KL_i = K_{\frac{i+1}{2}} || K'_{\frac{i+1}{2}+6}$, for $i = 1,3,5,7,9$; otherwise, $KL_i = K'_{\frac{i}{2}+2} || K_{\frac{i}{2}+4}$.

MISTY1 takes a 64-bit plaintext P as input and has a variable number of rounds; the officially recommended number of rounds is 8. The encryption algorithm $\mathtt{E}$ shown in Figure 3.1-(d) works as follows, where $L_0, R_0, \cdots, L_i, R_i$ are 32-bit variables, $KO_j = (KO_{j1} || KO_{j2} || KO_{j3} || KO_{j4})$ and $KI_j = (KI_{j1} || KI_{j2} || KI_{j3})$, $(j = 1, 2, \cdots, 8)$.

1) Set $(L_0 || R_0) = (P_L || P_R)$.

2) For $i = 1,3,5,7$, compute:

   - $R_i = \mathbf{FL}(L_{i-1}, KL_i)$, $L_i = \mathbf{FL}(R_{i-1}, KL_{i+1}) \oplus \mathbf{FO}(R_i, KO_i, KI_i)$,
   - $R_{i+1} = L_i$, $L_{i+1} = R_i \oplus \mathbf{FO}(L_i, KO_{i+1}, KI_{i+1})$.

3) Output ciphertext $C = \mathbf{FL}(R_8, KL_{10}) || \mathbf{FL}(L_8, KL_9)$.

We refer to the 8 rounds in the above description as Rounds $1, 2, \cdots, 8$, respectively.

3.3 A Related-Key Differential Attack of the Full MISTY1 under $2^{103.57}$ Weak Keys

In this section, we first review Dai and Chen's class of $2^{102.57}$ weak keys and their 7-round related-key differential characteristic with probability 2^{-58} under the class of weak keys. We then devise a related-key differential attack on the full MISTY1 when the secret key used is a weak key from the class of $2^{102.57}$ weak keys. Finally, we describe another class of $2^{102.57}$ weak keys under which similar results hold.

3.3.1 A Class of $2^{102.57}$ Weak Keys Owing to Dai and Chen

First, we define three constants which will be used subsequently: a 7-bit constant $a = 0010000$, a 16-bit constant $b = 0010000000010000$ and another 16-bit constant $c = 0010000000000000$ where all are represented in binary notation. Observe that $b = (a||0^2||a)$ and $c = (a||0^9)$, where 0^i represents a binary string of i zeros and so on for $i \geq 2$.

Let K_A and K_B be two 128-bit secret keys defined as follows:

- $K_A = (K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_8)$,
- $K_B = (K_1, K_2, K_3, K_4, K_5, K_6^*, K_7, K_8)$.

By the key schedule of MISTY1, we can get the corresponding eight 16-bit words for K_A and K_B which are denoted as follows:

- $K'_A = (K'_1, K'_2, K'_3, K'_4, K'_5, K'_6, K'_7, K'_8)$,
- $K'_B = (K'_1, K'_2, K'_3, K'_4, K'^*_5, K'^*_6, K'_7, K'_8)$.

Then, the class of weak keys is defined to be the set of all possible values of (K_A, K_B) that satisfy the following 12 conditions [3], where $K_{i,j}$ denotes the j-th bit of K_i. For instance, $K_{6,12}$ denotes the 12-th bit of K_6 and similar for $K_{7,3}, K_{7,12}, K_{8,3}, K'_{4,3}, K'_{4,12}$ and $K'_{7,3}$.

[3] Conditions (3.11) and (3.12) were based on the version of Dai and Chen's paper that we requested from Dai in February 2012 (Dai, 2012), where they used only the first ten conditions and obtained a class of 2^{105} weak keys. However, Dai and Chen had included these conditions in their formally published post-proceedings version (Dai & Chen, 2012).

$$K_6 \oplus K_6^* = c; \tag{3.1}$$

$$K_5' \oplus K_5'^* = b; \tag{3.2}$$

$$K_6' \oplus K_6'^* = c; \tag{3.3}$$

$$K_{6,12} = 0; \tag{3.4}$$

$$K_{7,3} = 1; \tag{3.5}$$

$$K_{7,12} = 0; \tag{3.6}$$

$$K_{8,3} = 1; \tag{3.7}$$

$$K_{4,3}' = 1; \tag{3.8}$$

$$K_{4,12}' = 1; \tag{3.9}$$

$$K_{7,3}' = 0; \tag{3.10}$$

$$\Pr_{\mathbf{FI}(\cdot,K_2')}(c \to c) > 0; \tag{3.11}$$

$$\Pr_{\mathbf{FI}(\cdot,K_7')}(b \to c) > 0. \tag{3.12}$$

Let us now analyse the number of weak keys. First, observe that when Condition (3.1) holds, then Condition (3.2) holds with certainty.

Note that $K_4' = \mathbf{FI}(K_4, K_5), K_6' = \mathbf{FI}(K_6, K_7)$ and $K_6'^* = \mathbf{FI}(K_6^*, K_7), K_7' = \mathbf{FI}(K_7, K_8)$. By performing a computer search, we get

- $|\{(K_2, K_3)|\text{Condition } (3.11)\}| = 2^{31}$;
- $|\{(K_4, K_5)|\text{Conditions } (3.2), (3.8) \ \& \ (3.9)\}| = 2^{30}$;
- $|\{(K_6, K_7, K_8)|\text{Conditions } (3.1), (3.3), (3.4), (3.5), (3.6), (3.7), (3.10) \ \& \ (3.12)\}| \approx 2^{25.57}$.

Note that there are 2^{16} possible values of K_1. Therefore, there are a total of approximately $2^{102.57}$ possible values of K_A satisfying Conditions (3.1)–(3.12) and thus

there are approximately $2^{102.57}$ weak keys. Furthermore, observe that $\Pr_{\mathbf{FI}(\cdot,K_7')}(b \to c)$ is 2^{-15} for each of the 9600 correct values of K_7', 2^{-14} for each of the 2432 correct values of K_7' and $\frac{6}{2^{16}} \approx 2^{-13.42}$ for each of the 128 correct values of K_7'.

3.3.2 Dai and Chen's 7-Round Related-Key Differential

Under the class of $2^{102.57}$ weak keys (K_A, K_B) described in Section 3.3.1, Dai and Chen presented the following 7-round related-key differential $\alpha \xrightarrow{7} \beta$: $(b||0^{32}||c) \to (0^{32}||c||0^{16})$ with probability 2^{-58} for Rounds 2–8. In Figure 3.2 and Figure 3.3, we illustrate the related-key differential characteristic in detail where $R_{4,3}$ denotes the 3-rd bit of R_4 (the right half of the output of Round 4) and $R_{4,12}$ denotes the 12-th bit of R_4.

As a result, Dai and Chen presented a related-key differential attack on 8-round MISTY1 without the first two FL functions by conducting a key recovery on $\mathbf{FO}_1$ in a way similar to the early abort technique for impossible differential cryptanalysis introduced by Lu et al. (2008).

3.3.3 Attacking the Full MISTY1 under the Class of $2^{102.57}$ Weak Keys

We find that the 7-round related-key differential with probability 2^{-58} can be used to conduct a related-key differential attack on the full MISTY1 when the secret key used is a weak key from the above described class of $2^{102.57}$ weak keys.

3.3.3 (a) Preliminary Results

First, we have the following result.

Proposition 3.1. *In the class of $2^{102.57}$ weak keys satisfying Conditions (3.1)–(3.12),*

1) *there are 2^{16} possible values of each of K_1, K_3 and K_5;*

2) *there are $2^{25.57}$ possible values of (K_6, K_7, K_8); in particular, there are a total of $2^{13.57}$ possible values of K_7' and for every possible such value, there are 2^{12}*

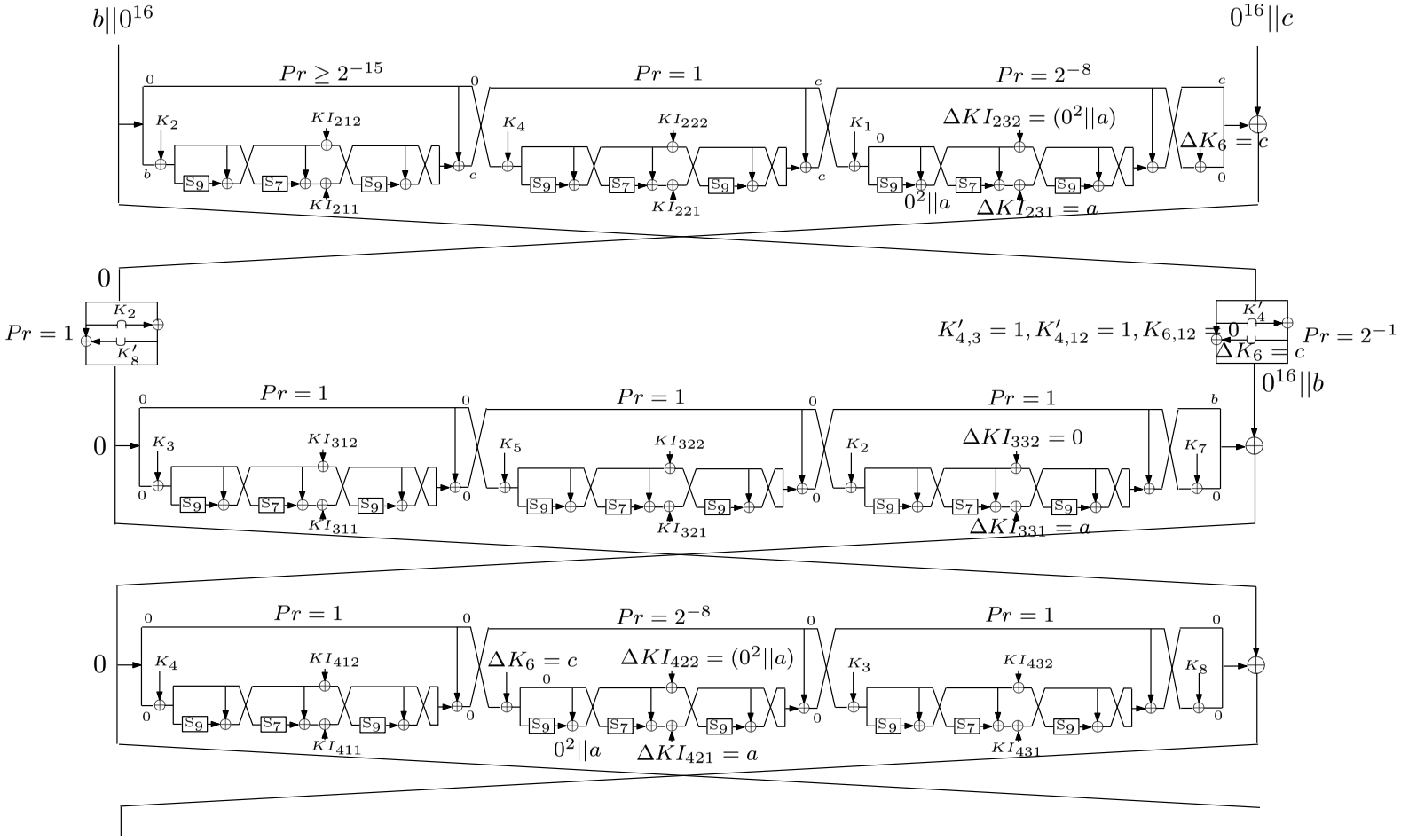


Figure 3.2: Chen and Dai's Related-Key Differential for Rounds 2–4

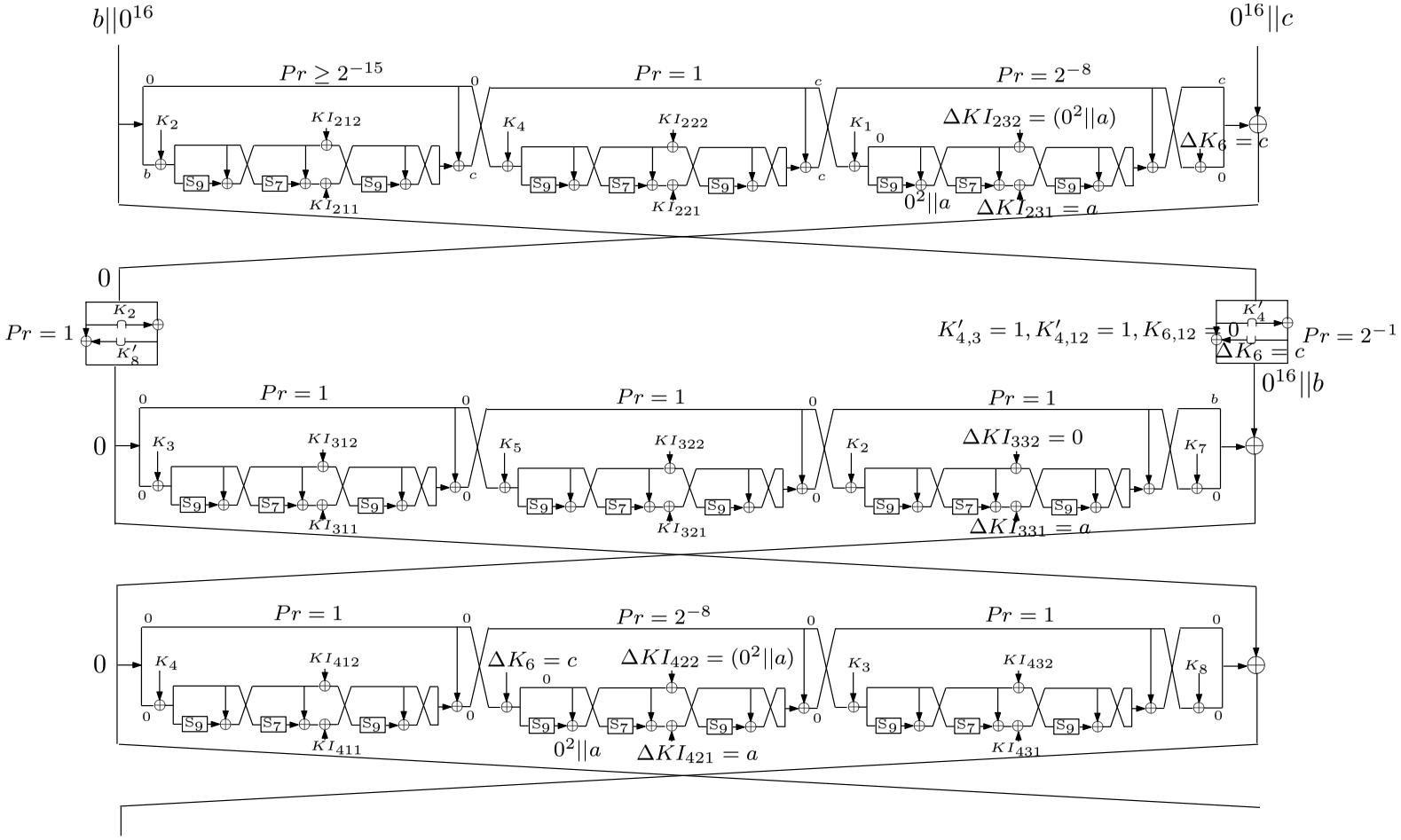


Figure 3.3: Chen and Dai's Related-Key Differential for Rounds 5–8

possible values of (K'_6, K_8)*;*

3) *there are a total of* 2^8 *possible values of* $K'_{2,8-16}$*,* 2^{16} *possible values of* K'_3 *and* 2^8 *possible values of* $K'_{4,8-16}$*, where* $K'_{2,8-16}$ *denotes bits* $(8,\cdots,16)$ *of* K'_2 *and* $K'_{4,8-16}$ *denotes bits* $(8,\cdots,16)$ *of* K'_4*;*

4) $\Pr_{\mathbf{FI}(\cdot,\forall K'_7)}(b \to c) \geq 2^{-15}$ *and* $\Pr_{\mathbf{FI}(\cdot,\forall K'_2)}(c \to c) = 2^{-15}$.

Next, we concentrate on the propagation of the input difference α (i.e., $b||0^{32}||c$) of the 7-round differential through the preceding Round 1 which includes the $\mathbf{FL}_1$ and $\mathbf{FL}_2$ functions under (K_A, K_B); see Figure 3.4.

Under (K_A, K_B), by the key schedule of MISTY1, we have

- $\Delta KO_{11} = \Delta K_1 = 0, \Delta KO_{12} = \Delta K_3 = 0,$
- $\Delta KO_{13} = \Delta K_8 = 0, \Delta KO_{14} = \Delta K_5 = 0,$
- $\Delta KI_{11} = \Delta K'_6 = c,$
- $\Delta KI_{12} = \Delta K'_2 = 0,$
- $\Delta KI_{13} = \Delta K'_4 = 0,$
- $\Delta KL_1 = \Delta(K_1||K'_7) = 0,$
- $\Delta KL_2 = \Delta(K'_3||K_5) = 0.$

As depicted in Figure 3.4, the right half of α is $(0^{16}||c)$, so the $\mathbf{FI}_{11}$ function has a zero input difference. Since $\Delta KO_{11} = 0$ and $\Delta KI_{11} = c$, the output difference of $\mathbf{FI}_{11}$ is b with probability of one. The input difference of the $\mathbf{FI}_{12}$ function is c, and thus the first S_9 function in $\mathbf{FI}_{12}$ has an input difference $a||0^2$ and we assume its output difference is $A \in \{0,1\}^9$. In addition, the S_7 function in $\mathbf{FI}_{12}$ has a zero input and output difference. The second S_9 function in $\mathbf{FI}_{12}$ has an input difference A and we assume its output difference is $B \in \{0,1\}^9$. As a result, the $\mathbf{FI}_{12}$ function

has an output difference $X = (\text{Trunc}(A)||(B \oplus (0^2||\text{Trunc}(A))))$. A simple computer simulation reveals that $\text{Trunc}(A)$ can take all 2^7 possible values and thus we assume that X can take all values in $\{0,1\}^{16}$.

Since the input difference of the $\mathbf{FI}_{13}$ function is $0^9||a$, the first S_9 function in $\mathbf{FI}_{13}$ has a zero input difference. The S_7 function in $\mathbf{FI}_{13}$ has an input difference a and we assume its output difference is $D \in \{0,1\}^7$, which can take only 2^6 possible values. The second S_9 function in $\mathbf{FI}_{13}$ has an input difference $0^2||a$ and we assume its output difference is $E \in \{0,1\}^9$. Consequently, the $\mathbf{FI}_{13}$ function has an output difference $Y = ((a \oplus D)||(E \oplus (0^2||(a \oplus D))))$ and it can take about 2^{15} values in $\{0,1\}^{16}$. We denote the set of 2^{15} values by $\mathscr{S}_d$.

The $\mathbf{FL}_1$ function has an output difference $(0^{16}||c)$, so its input difference can only be of the form $\overbrace{00?0000000000000}^{32\ bits}||00?0000000000000$ which will be denoted by $\eta = (\eta_L, \eta_R)$ in the following descriptions where the question marker "?" represents an indeterminate bit. However, when the first question marker takes a zero value, the second question marker can take only 1, that is, η has only three possible values (the specific form depends on the values of the two subkey bits $K_{1,3}$ and $K'_{7,3}$). The $\mathbf{FL}_2$ function has an output difference $(X \oplus c)||(X \oplus Y \oplus (0^9||a))$, so its input difference is indeterminate, denoted by "?" in Figure 3.4.

From the above analysis, we conclude that the subkeys KI_{121} and KI_{131} do not affect the values of X and Y and thus they are not required when checking whether a candidate plaintext pair generates the input difference $\alpha = (b||0^{32}||c)$ of the 7-round related-key differential. Further, as $K'_3 = \mathbf{FI}(K_3, K_4), K'_4 = \mathbf{FI}(K_4, K_5), K'_6 = \mathbf{FI}(K_6, K_7)$ and $K'_7 = \mathbf{FI}(K_7, K_8)$, we obtain the following result.

Proposition 3.2. *Only the subkeys* $(K_1, K'_{2,8-16}, K_3, K_4, K_5, K_6, K_7, K_8)$ *are required when checking whether a candidate plaintext pair produces the input difference* $\alpha = (b||0^{32}||c)$ *of the 7-round related-key differential.*

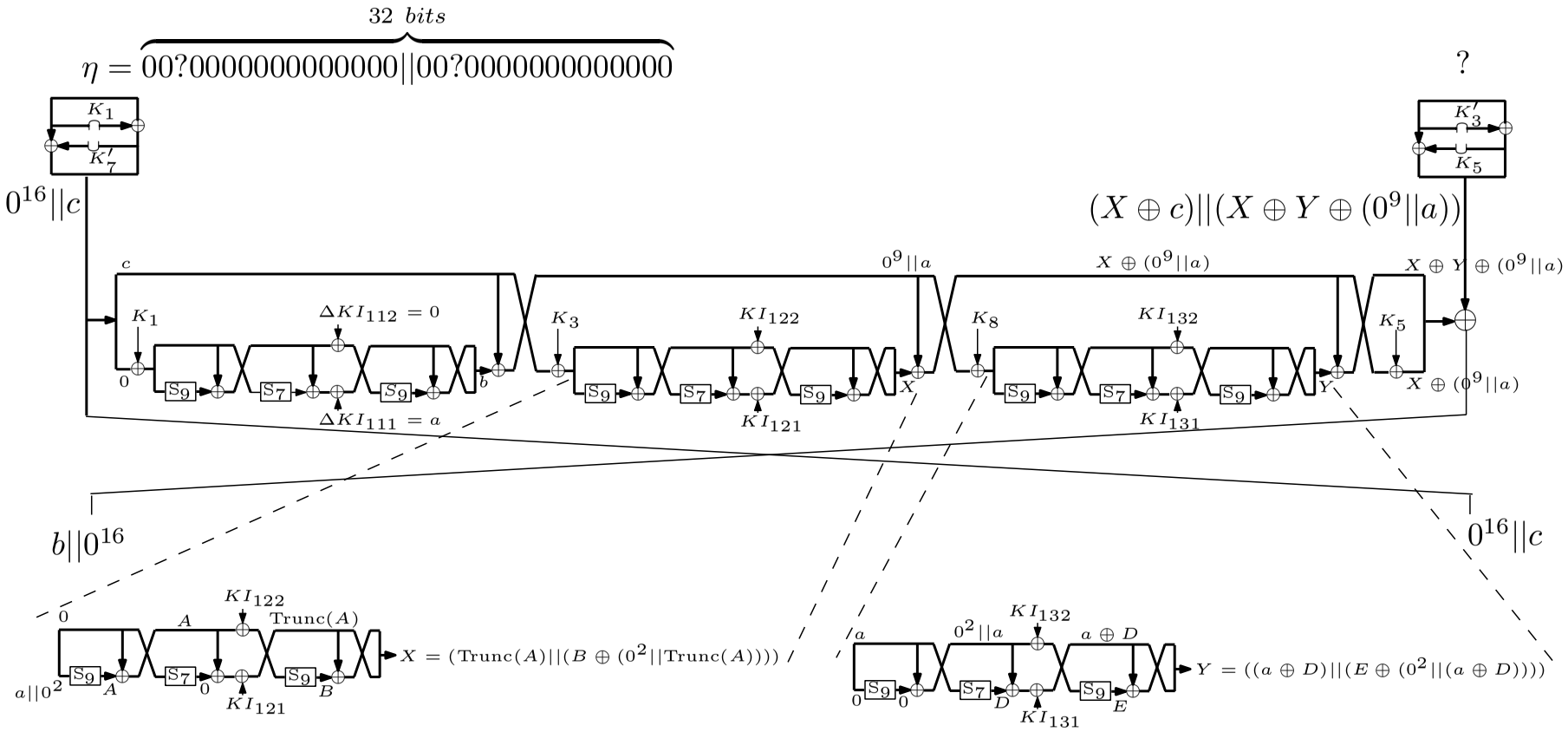


Figure 3.4: Propagation of α through the Inverse of Round 1 with FL_1 and FL_2

3.3.3 (b) *Attack Procedure*

First, we precompute two hash tables $\mathscr{T}_1$ and $\mathscr{T}_2$. Observe that from the left halves of a pair of plaintexts, we only need $(K_1, K_3, K'_{2,8-16})$ when computing the output difference X of the $\mathbf{FI}_{12}$ function and only need $(K_1, K'_6, K'_7, K_8, K'_{4,8-16})$ when computing the output difference Y of the $\mathbf{FI}_{13}$ function. To generate $\mathscr{T}_1$ and $\mathscr{T}_2$, we do the following procedure under every 32-bit value $x = (x_L||x_R)$.

1) For every possible K_1:
   a) Compute $Z = (x_L \cap K_1) \oplus ((x_L \oplus \eta_L) \cap K_1) \oplus \eta_R$ and proceed to the following steps only when $Z = c$.
   b) For every possible $(K_3, K'_{2,8-16})$, compute the output difference of $\mathbf{FI}_{12}$ as X.
2) Store all satisfying $(K_1, K_3, K'_{2,8-16})$ into Table $\mathscr{T}_1$ indexed by (x, η, X).
3) For every possible K'_7:
   a) Compute $W = \eta_L \oplus (((x_L \cap K_1) \oplus x_R) \cup K'_7) \oplus (((x_L \cap K_1) \oplus x_R \oplus c) \cup K'_7)$ and proceed to the following steps only when $W = 0$.
   b) For every possible $(K'_6, K_8, K'_{4,8-16})$, compute the output difference of $\mathbf{FI}_{13}$ as Y.
4) Store the values of (K_6, K_7, K_8) corresponding to all satisfying values of (K'_6, K'_7, K_8) into Table $\mathscr{T}_2$ indexed by $(x, \eta, Y, K_1, K'_{4,8-16})$.

There are 2^{16} possible values of K_1, 2^{16} possible values of K_3, 2^8 possible values of $K'_{2,8-16}$ and 3 possible values of η. For a fixed (x, η, X), there are $2^{16} \times 2^{-1} \times 2^{16} \times 2^8 \times 2^{-16} = 2^{23}$ satisfying values of $(K_1, K_3, K'_{2,8-16})$ on average in $\mathscr{T}_1$. The precomputation for $\mathscr{T}_1$ takes about $2^{32} \times 3 \times 2^{16} \times 2^{16} \times 2^8 \approx 2^{73.59}$ **FI** computations and $\mathscr{T}_1$ requires a memory of about $2^{24} \times 2^{32} \times 3 \times 2^{16} \times \frac{16+16+8}{8} \approx 2^{75.91}$ bytes. There are $2^{13.57}$ possible values of K'_7, 2^{12} possible values of (K'_6, K_8), 2^8 possible values of

$K'_{4,8-16}$ and 2^{15} possible values of Y. For a fixed $(x,\eta,Y,K_1,K'_{4,8-16})$, on average there are $2^{13.57}\times 2^{-1}\times 2^{12}\times 2^{-15}=2^{9.57}$ satisfying values of (K'_6,K'_7,K_8) in $\mathscr{T}_2$. The precomputation for $\mathscr{T}_2$ takes about $2^{32}\times 3\times 2^{16}\times 2^{13.57}\times 2^{12}\times 2^8\times 2\approx 2^{84.16}$ **FI** computations and $\mathscr{T}_2$ requires a memory of about $2^{9.57}\times 2^{32}\times 3\times 2^{15}\times 2^{16}\times 2^8\times 6\approx 2^{84.74}$ bytes. Note that we can use several tricks to optimise the procedure to reduce the computational complexity for generating the two tables. However, it is negligible compared to the computational complexity of the following online attack procedure.

We devise the following attack procedure to break the full MISTY1 when a weak key is used.

1) Initialise zero to an array of $2^{95.57}$ counters corresponding to all the $2^{95.57}$ possible values of $(K_1,K'_{2,8-16},K_3,K_4,K_5,K_6,K_7,K_8)$.

2) Choose 2^{60} ciphertext pairs $(C,C^*=C\oplus(0^{32}||c||0^{16}))$. In a chosen ciphertext attack scenario, obtain the plaintexts for the ciphertexts C,C^* under K_A,K_B respectively. We denote the plaintext for ciphertext C encrypted under K_A by $P=(PL_L||PL_R,PR_L||PR_R)$ and the plaintext for ciphertext C^* encrypted under K_B by $P^*=(PL^*_L||PL^*_R,PR^*_L||PR^*_R)$.

3) Check whether a plaintext pair (P,P^*) meets the condition $(PL_L||PL_R)\oplus(PL^*_L||PL^*_R)=\eta$ by first checking the 30 bit positions with a zero difference and then checking the remaining two bit positions. Keep only the satisfying plaintext pairs.

4) For every remaining plaintext pair (P,P^*), do the following sub-steps.

   a) Guess a possible value of (K'_3,K_5) and compute (X,Y) such that $(X\oplus c)||(X\oplus Y\oplus(0^9||a))=\mathbf{FL}(PR_L||PR_R,K'_3||K_5)\oplus\mathbf{FL}(PR^*_L||PR^*_R,K'_3||K_5)$. Execute the next steps only if $Y\in\mathscr{S}_d$; otherwise, repeat this step with another subkey guess.

   b) Access Table $\mathscr{T}_1$ at entry $(PL_L||PL_R,\eta,X)$ to get the satisfying values of $(K_1,K_3,K'_{2,8-16})$.

c) For each satisfying value of $(K_1, K_3, K'_{2,8-16})$, retrieve K_4 from the equation $K'_3 = \mathbf{FI}(K_3, K_4)$, compute $K'_4 = \mathbf{FI}(K_4, K_5)$ and access Table $\mathcal{T}_2$ at entry $(PL_L||PL_R, \eta, Y, K_1, K'_{4,8-16})$ to get the satisfying values of (K_6, K_7, K_8).

d) Increase 1 to each of the counters corresponding to the obtained values of $(K_1, K'_{2,8-16}, K_3, K_4, K_5, K_6, K_7, K_8)$.

5) For a value of $(K_1, K'_{2,8-16}, K_3, K_4, K_5, K_6, K_7, K_8)$ whose counter number is equal to or larger than 3, exhaustively search the remaining 7 key bits with two known plaintext-ciphertext pairs. If a value of $(K_1, K_2, \cdots, K_8)$ is suggested, output it as the secret key of the full MISTY1.

3.3.3 (c) Attack Complexity

The attack requires $2^{60} \times 2 = 2^{61}$ chosen ciphertexts. In Step 3, only $2^{60} \times 2^{-30} \times \frac{3}{4} \approx 2^{29.58}$ plaintext pairs are expected to satisfy the condition and it takes about 2^{60} memory accesses to obtain the satisfying plaintext pairs. Step 4(a) has a time complexity of about $2^{29.58} \times 2^{16} \times 2^{16} \times 2 = 2^{62.58}$ **FL** computations. In Step 4(b), for a plaintext pair and a possible value of (K'_3, K_5), we obtain, on average, 2^{23} possible values of $(K_1, K_3, K'_{2,8-16})$ as discussed in the precomputation phase. Owing to the filtering condition in Step 4(a), Step 4(b) has a time complexity of about $2^{29.58} \times \frac{2^{15}}{2^{16}} \times 2^{32} \times 2^{23} = 2^{83.58}$ memory accesses (if conducted on a 64-bit computer). In Step 4(c), for a plaintext pair and a possible value of $(K_1, K_3, K_5, K'_{2,8-16}, K'_3)$, we obtain, on average, $2^{9.57}$ possible values of (K_6, K_7, K_8) as discussed in the precomputation phase, and thus Step 4(c) has a time complexity of about $2^{28.58} \times 2^{32} \times 2^{23} \times 2^{9.57} = 2^{93.15}$ memory accesses. Step 4(d) has a time complexity of about $2^{93.15} \times 2 = 2^{94.15}$ memory accesses where the factor "2" represents that it requires two memory accesses for a single access to an entry whose length is between 65 and 128 bits when conducted on a 64-bit computer.

The probability that the counter for a wrong $(K_1, K'_{2,8-16}, K_3, K_4, K_5, K_6, K_7, K_8)$ has a number larger than or equal to 3 is approximately $\sum_{i=3}^{2^{60}} [\binom{2^{60}}{i} \cdot (2^{-64})^i \cdot (1 - 2^{-64})^{2^{60}-i}] \approx 2^{-14.67}$. Thus, it is expected that there are a total of $2^{95.57} \times 2^{-14.67} =$

$2^{80.9}$ wrong values of $(K_1, K'_{2,8-16}, K_3, K_4, K_5, K_6, K_7, K_8)$ whose counters have a number greater than or equal to 3. Thus, it requires $2^{80.9} \times 2^7 + 2^{80.9} \times 2^7 \times 2^{-64} \approx 2^{87.9}$ trial encryptions to check them in Step 5. In Step 5, a wrong value of $(K_1, K_2, \cdots, K_8)$ is suggested with probability $2^{-64\times 2} = 2^{-128}$, so the number of suggested values of $(K_1, K_2, \cdots, K_8)$ is expected to be $2^{87.9} \times 2^{-128} = 2^{-40.1}$ which is rather low. Thus, the time complexity of the attack is dominated by Steps 4(c), 4(d) and 5. An extremely conservative estimate is: 16 memory accesses equal a full MISTY1 encryption in terms of time, assuming that in every round, say Round i, the $\mathbf{FI}_{i1}$ and $\mathbf{FI}_{i2}$ functions that are implemented in parallel are equivalent to one memory access and the subsequent $\mathbf{FI}_{i3}$ function is equivalent to another one memory access by neglecting the key schedule algorithm and the computational complexities of other operations. Thus, one round is equivalent to 2 memory accesses[4]. Therefore, the attack has a total time complexity of about $\frac{2^{93.15}+2^{94.15}}{16} + 2^{87.9} \approx 2^{90.93}$ MISTY1 encryptions.

The counter for the correct key has an expected number of $2^{60} \times 2^{-58} = 4$ and the probability that the counter for the correct key has a number at least 3 is approximately $\sum_{i=3}^{2^{60}} [\binom{2^{60}}{i} \cdot (2^{-58})^i \cdot (1-2^{-58})^{2^{60}-i}] \approx 0.76$. Therefore, the related-key differential attack has a success probability of 76%. The memory complexity of the attack is dominated by the space for the array of $2^{95.57}$ counters, which is $2^{95.57} \times \frac{95.57}{8} \approx 2^{99.2}$ bytes. It is worthy to note that there exist time–memory tradeoff versions to the above attack.

3.3.4 Another Class of $2^{102.57}$ Weak Keys

In the above sub-sections, we have described a class of $2^{102.57}$ weak keys and a related-key differential attack on the full MISTY1 under a weak key. However, we observe that there exists another class of $2^{102.57}$ weak keys under which similar results hold. The new weak key class is obtained by setting $K'_{7,3} = 1$, which is further classified

[4]The question on how many memory accesses (table lookups) are equivalent to one MISTY1 encryption in terms of time depends closely on the used platform and MISTY1 implementation as well as the storage location of the hash table. In theoretical block cipher cryptanalysis, it is usually assumed by default that a hash table is stored in an ideal place, RAM say, like an S-box table; and it takes an almost constant time to access an entry in a hash table, independent of the number of entries.

into two sub-classes by the possible values of the subkey bit $K_{1,3}$. This will affect only the $\mathbf{FL}_{10}$ function in the 7-round related-key differential, but the output difference of $\mathbf{FL}_{10}$ will be fixed once $K_{1,3}$ is given, that is, the right half of the output difference of the resulting 7-round related-key differential will be $c||c$ when $K_{1,3} = 1$ and $0^{16}||c$ when $K_{1,3} = 0$. Thus, by choosing a number of ciphertext pairs with a corresponding difference, we can conduct a similar attack on the full MISTY1 under every sub-class of weak keys. In total, we have $2^{103.57}$ weak keys under which a related-key differential attack can break the full MISTY1.

3.4 A Related-Key Amplified Boomerang Attack of the Full MISTY1 under 2^{92} Weak Keys

In this section, we first review Chen and Dai's class of 2^{90} weak keys and their 7-round related-key amplified boomerang distinguisher with probability 2^{-118}. Next, we describe a slight improvement to Chen and Dai's 7-round related-key amplified boomerang distinguisher which has a probability of 2^{-116} and then present a related-key amplified boomerang attack on the full MISTY1 under the class of 2^{90} weak keys. Finally, we describe three other classes of 2^{90} weak keys under which there exist similar results.

3.4.1 A Class of 2^{90} Weak Keys Owing to Chen and Dai

First, we define the same three constants a, b and c as used in Section 3.3.1, that is, a 7-bit constant $a = 0010000$, a 16-bit constant $b = 0010000000010000$ and another 16-bit constant $c = 0010000000000000$ where all are represented in binary notation.

Let K_A, K_B, K_C and K_D be four 128-bit secret keys defined as follows:

- $K_A = (K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_8)$,
- $K_B = (K_1, K_2^*, K_3, K_4, K_5, K_6, K_7, K_8)$,

- $K_C = (K_1, K_2, K_3, K_4, K_5, K_6^*, K_7, K_8)$,
- $K_D = (K_1, K_2^*, K_3, K_4, K_5, K_6^*, K_7, K_8)$.

By the key schedule of MISTY1, we can get the corresponding eight 16-bit words for K_A, K_B, K_C and K_D which are denoted as follows:

- $K'_A = (K'_1, K'_2, K'_3, K'_4, K'_5, K'_6, K'_7, K'_8)$,
- $K'_B = (K'^*_1, K'^*_2, K'_3, K'_4, K'_5, K'_6, K'_7, K'_8)$,
- $K'_C = (K'_1, K'_2, K'_3, K'_4, K'^*_5, K'^*_6, K'_7, K'_8)$,
- $K'_D = (K'^*_1, K'^*_2, K'_3, K'_4, K'^*_5, K'^*_6, K'_7, K'_8)$.

The class of weak keys is defined to be the set of all possible values of (K_A, K_B, K_C, K_D) that satisfy the following 12 conditions where $K_{i,j}$ denotes the j-th bit of K_i. For instance, $K_{5,3}$ denotes the 3-rd bit of K_5 and similarly for $K_{5,12}, K'_{4,3}, K_{7,3}, K_{7,12}$, and $K_{8,3}$.

$$K_2 \oplus K_2^* = c; \tag{3.13}$$

$$K_6 \oplus K_6^* = c; \tag{3.14}$$

$$K_1' \oplus K_1'^* = b; \tag{3.15}$$

$$K_5' \oplus K_5'^* = b; \tag{3.16}$$

$$K_2' \oplus K_2'^* = c; \tag{3.17}$$

$$K_6' \oplus K_6'^* = c; \tag{3.18}$$

$$K_{5,3} = 1; \tag{3.19}$$

$$K_{5,12} = 0; \tag{3.20}$$

$$K_{4,3}' = 0; \tag{3.21}$$

$$K_{7,3} = 1; \tag{3.22}$$

$$K_{7,12} = 0; \tag{3.23}$$

$$K_{8,3} = 0. \tag{3.24}$$

Let us now analyse the number of the weak keys. First observe that when Condition (3.13) holds, then Condition (3.15) holds with certainty. Besides, when Condition (3.14) holds, Condition (3.16) holds with certainty.

Note that $K_2' = \mathbf{FI}(K_2, K_3), K_2'^* = \mathbf{FI}(K_2^*, K_3), K_4' = \mathbf{FI}(K_4, K_5), K_6' = \mathbf{FI}(K_6, K_7)$ and $K_6'^* = \mathbf{FI}(K_6^*, K_7)$. By performing a computer search, we get

- $|\{(K_2, K_3)|\text{Conditions } (3.13) \text{ and } (3.17)\}| = 2^{16}$;
- $|\{(K_4, K_5)|\text{Conditions } (3.19), (3.20) \text{ and } (3.21)\}| = 2^{29}$;
- $|\{(K_6, K_7)|\text{Conditions } (3.14), (3.18), (3.22) \text{ and } (3.23)\}| = 2^{14}$.

Note that there are 2^{16} and 2^{15} possible values of K_1 and K_8 respectively. Therefore, Chen and Dai (2011) claimed that there are a total of 2^{90} possible values of K_A

satisfying the above 12 conditions and thus there are 2^{90} weak keys.

3.4.2 Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher

Now, we describe Chen and Dai's related-key amplified boomerang distinguisher for Rounds 1–7 under the class of 2^{90} weak keys (K_A, K_B, K_C, K_D) described in Section 3.4.1.

The first related-key differential $\alpha \xrightarrow{2} \beta$ for this distinguisher is the 2-round related-key differential $(0^{48}||b) \rightarrow (0^{32}||c||0^{16})$ with probability of one for Rounds 1–2 under (K_A, K_B) or under (K_C, K_D), where 0^i represents a binary string of i zeros for $i \geq 2$. The second related-key differential $\gamma \xrightarrow{5} \delta$ for this distinguisher is the 5-round related-key differential $(0^{48}||b) \rightarrow 0$ with probability 2^{-27} for Rounds 3–7 under (K_A, K_C) or under (K_B, K_D). In Figure 3.5, Figure 3.6 and Figure 3.7, we illustrate the two related-key differentials in detail where $R_{4,3}$ denotes the 3-rd bit of R_4 (the right half of the output of Round 4) and $R_{4,12}$ denotes the 12-th bit of R_4.

As a result, Chen and Dai obtained a 7-round related-key amplified boomerang distinguisher with probability $1^2 \times (2^{-27})^2 \times 2^{-64} = 2^{-118}$ under a weak key (K_A, K_B, K_C, K_D). As a result, they presented an attack on 8-round MISTY1 without the last two FL functions by conducting a key recovery on $\mathbf{FO}_8$ in a way similar to the early abort technique (Lu et al., 2008).

3.4.3 An Improved 7-Round Related-Key Amplified Boomerang Distinguisher

We first focus on the $\mathbf{FI}_{73}$ function in the second 5-round related-key differential $\gamma \xrightarrow{5} \delta$ used in Chen and Dai's 7-round distinguisher where the probability is 2^{-16}. Observe that $KI_{73} = K_2'$ or $K_2'^*$, depending on which pair from a quartet is considered. Chen and Dai used a probability value of 2^{-16} for the differential $c \rightarrow c$ operating on $\mathbf{FI}_{73}$ (an alternative explanation is to consider the two S_9 S-boxes where each having a probability value of 2^{-8}). However, intuitively we should make sure that a weak key (K_A, K_B, K_C, K_D) should also satisfy the condition that the differential $c \rightarrow c$ is a possible differential for $\mathbf{FI}_{73}$; otherwise, the differential $c \rightarrow c$ would have a zero probability

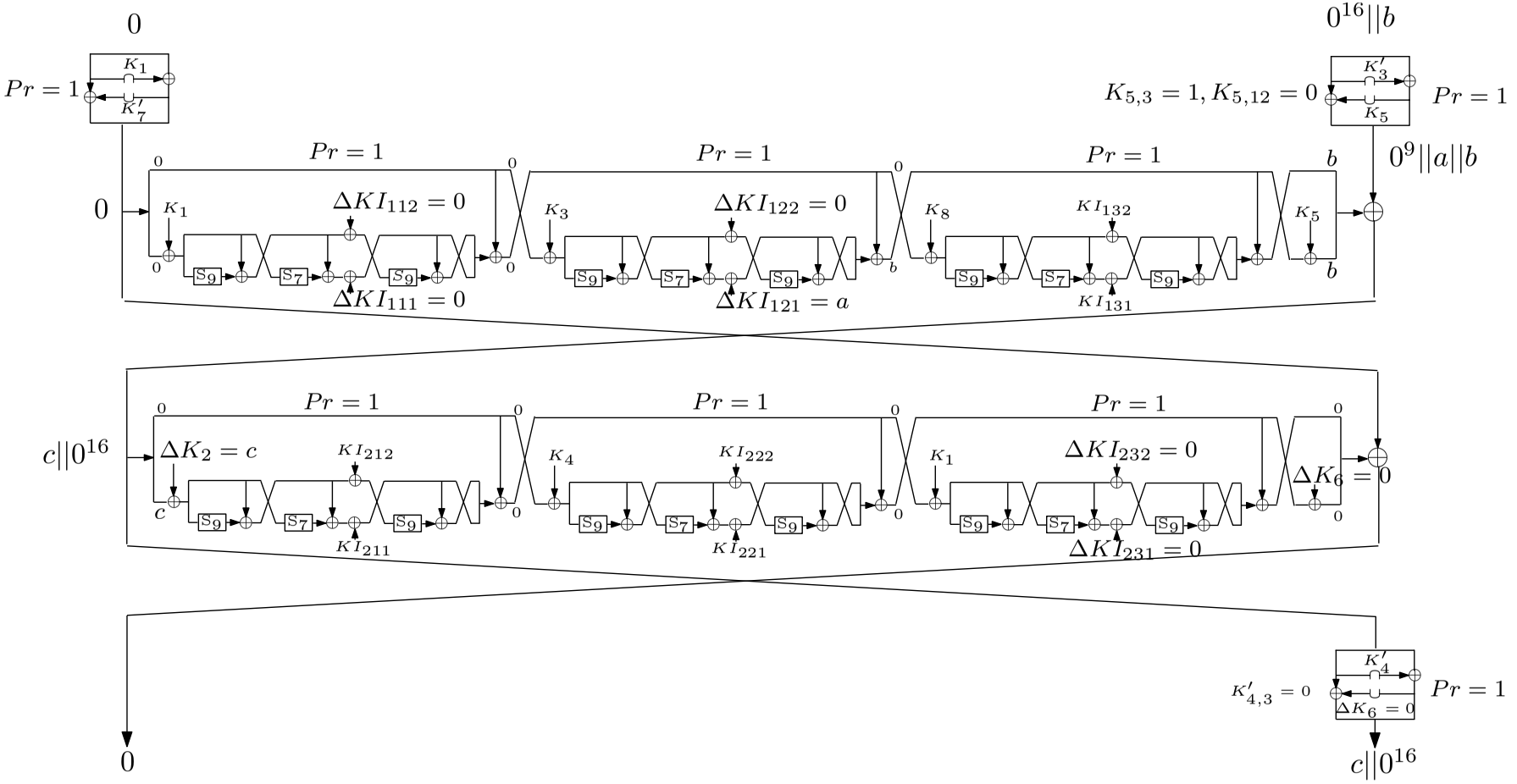


Figure 3.5: Thet Related-key Differential for Rounds 1-2 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher

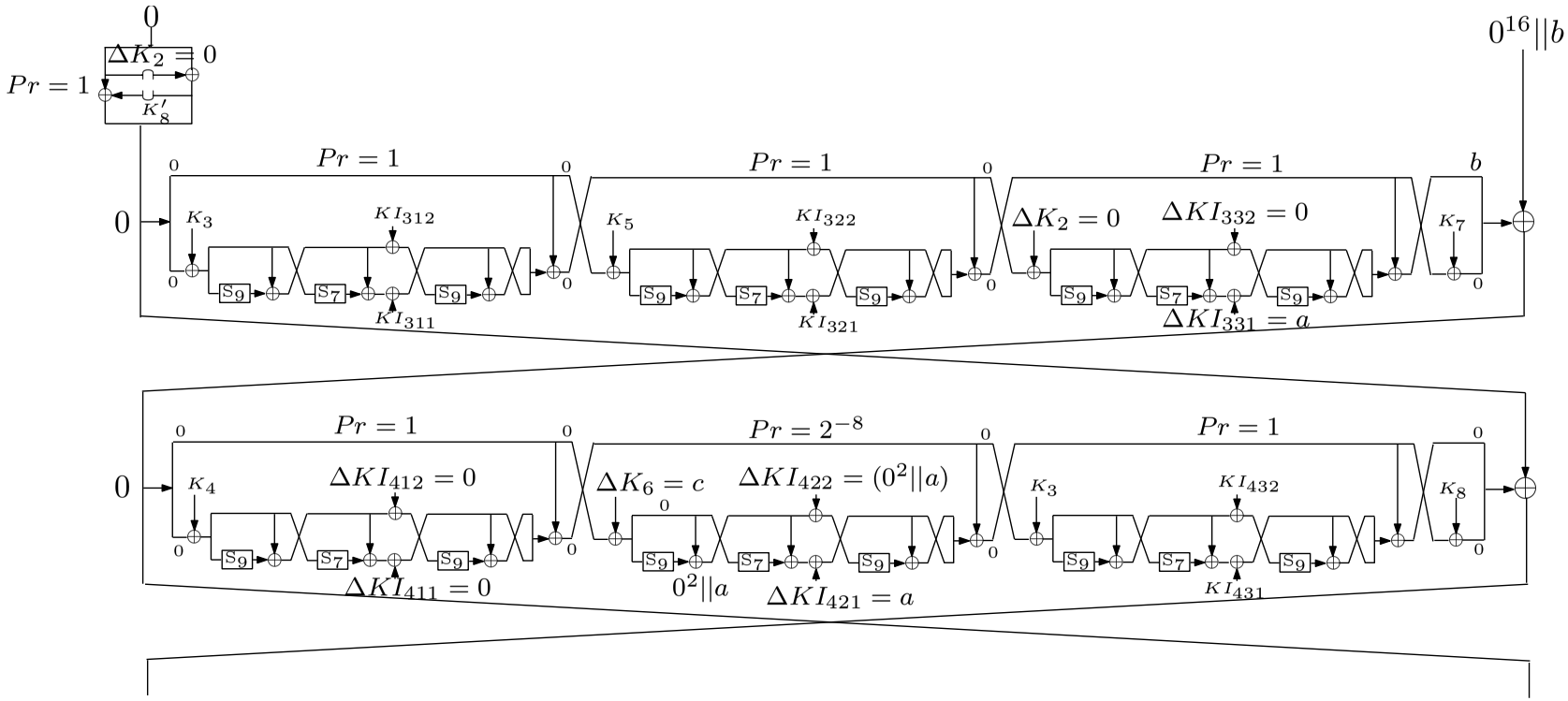


Figure 3.6: Thet Related-key Differential for Rounds 3-4 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher

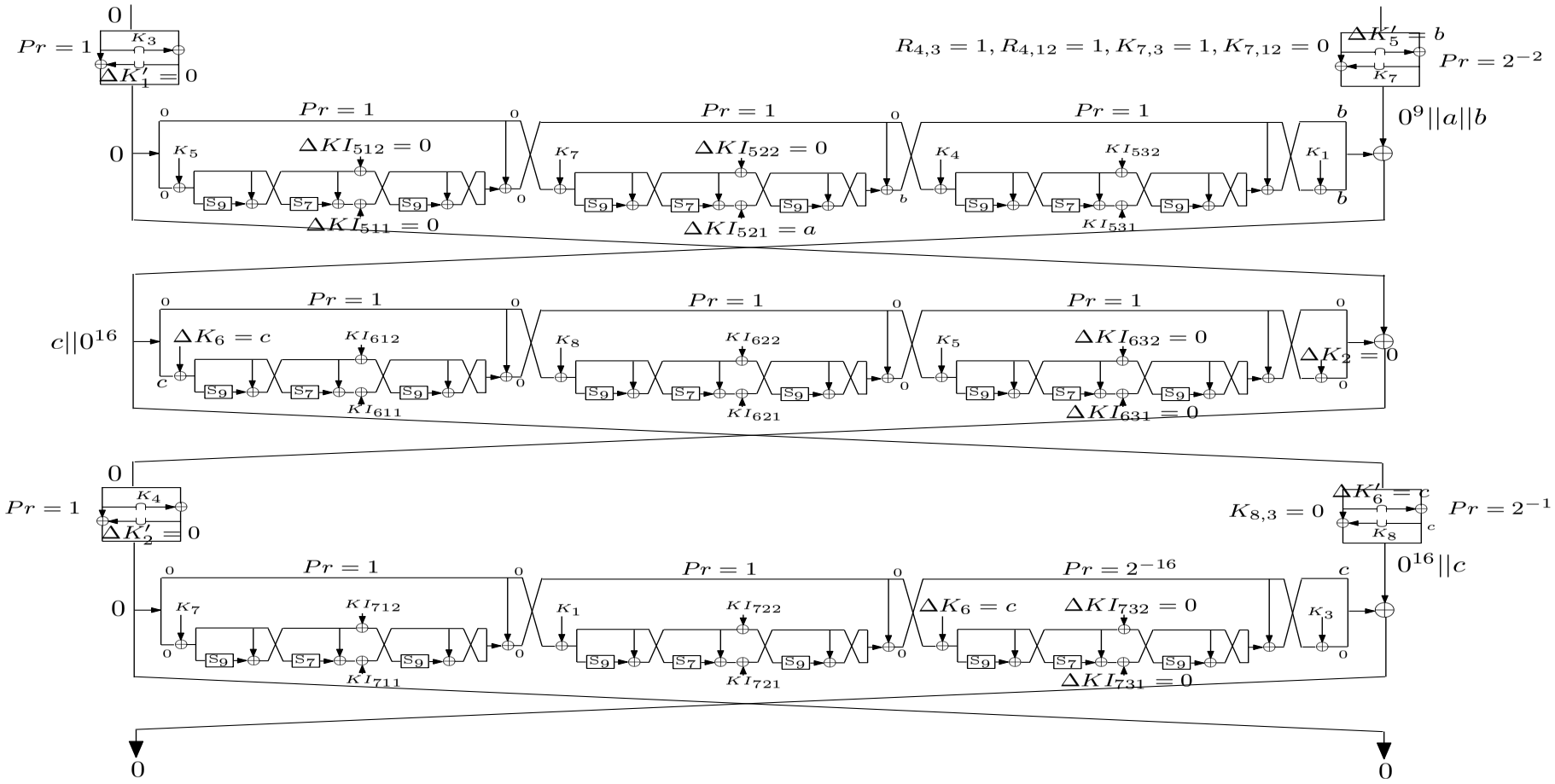


Figure 3.7: Thet Related-key Differential for Rounds 5-7 Used in Chen and Dai's 7-Round Related-Key Amplified Boomerang Distinguisher

and the 7-round distinguisher would be flawed. Thus, we should put the following two additional conditions when defining a set of weak keys:

$$\Pr_{\mathbf{FI}(\cdot,K_2')}(c \to c) > 0; \tag{3.25}$$

$$\Pr_{\mathbf{FI}(\cdot,K_2'^*)}(c \to c) > 0. \tag{3.26}$$

After performing a computer program, we surprisingly found that the number of (K_2, K_3) satisfying Conditions (3.13), (3.17), (3.25) and (3.26) is equal to the number of (K_2, K_3) satisfying Conditions (3.13) and (3.17), that is $|\{(K_2, K_3)|\text{Conditions } (3.13), (3.17), (3.25) \text{ and } (3.26)\}| = 2^{16}$. This means that the class of weak keys satisfying Conditions (3.13)–(3.26) is the same as the class of weak keys satisfying Conditions (3.13)–(3.24) owing to Chen and Dai. Nevertheless, we find something valuable. For each possible K_2' or $K_2'^*$, there are exactly two pairs of inputs to $\mathbf{FI}_{73}$ which follow the differential $c \to c$, that is to say, the differential $c \to c$ for $\mathbf{FI}_{73}$ has a probability of 2^{-15} which is twice as large as the probability value used by Chen and Dai.

Therefore, the second 5-round related-key differential $\gamma \xrightarrow{5} \delta$ used in Chen and Dai's 7-round distinguisher actually has a probability of 2^{-26} and the resulting 7-round distinguisher has probability $1^2 \times (2^{-26})^2 \times 2^{-64} = 2^{-116}$ under a weak key (K_A, K_B, K_C, K_D).

In particular, we have the following result.

Proposition 3.3. *In the class of 2^{90} weak keys satisfying Conditions (3.13)–(3.26),*

1) *there are 2^{16} possible values of K_1, 2^{14} possible values of K_5 and 2^{15} possible values of K_8;*

2) *there are 2^{14} possible values of (K_6, K_7); in particular, there are a total of 2^{13} possible values of K_7 and for every possible such value, there are 2 possible values of K_6;*

3) there are a total of 2^{16} *possible values of* K_3'*;*

4) $\Pr_{\mathbf{FI}(\cdot,\forall K_2')}(c \to c) = \Pr_{\mathbf{FI}(\cdot,\forall K_2'^*)}(c \to c) = 2^{-15}$.

3.4.4 Attacking the Full MISTY1 under the Class of 2^{90} Weak Keys

We devise a related-key amplified boomerang attack on the full MISTY1 under a weak key from the weak key class based on the 7-round related-key amplified boomerang distinguisher with probability 2^{-116}.

3.4.4 (a) Preliminary Results

First concentrate on the propagation of the output difference δ (i.e., 0) of the 7-round distinguisher through the following Round 8 which includes the $\mathbf{FL}_9$ and $\mathbf{FL}_{10}$ functions under (K_A, K_C) or under (K_B, K_D); see Figure 3.8.

Under (K_A, K_C), by the key schedule of MISTY1, we have

- $\Delta KO_{81} = \Delta K_8 = 0$,
- $\Delta KO_{82} = \Delta K_2 = 0$,
- $\Delta KO_{83} = \Delta K_7 = 0$,
- $\Delta KO_{84} = \Delta K_4 = 0$,
- $\Delta KI_{81} = \Delta K_5' = b$,
- $\Delta KI_{82} = \Delta K_1' = 0$,
- $\Delta KI_{83} = \Delta K_3' = 0$,
- $\Delta KL_9 = \Delta(K_5 || K_3') = 0$,
- $\Delta KL_{10} = \Delta(K_7' || K_1) = 0$.

Since $\delta = 0$, both the $\mathbf{FI}_{81}$ and $\mathbf{FI}_{82}$ functions have a zero input difference. The first S_9 and S_7 in $\mathbf{FI}_{81}$ both have a zero input difference. However, as $\Delta KI_{81} = b$, it follows that the second S_9 in $\mathbf{FI}_{81}$ has an input difference $0^2||a$, and thus the output difference of the $\mathbf{FI}_{81}$ function has a form of $a||X$ where $X \in \{0,1\}^9$ can take only 2^8 possible values and we denote by $\mathscr{S}_a$ the set of the 2^8 possible values of X. Since $\Delta KO_{82} = 0$ and $\Delta KI_{82} = 0$, the $\mathbf{FI}_{82}$ function has a zero output difference. Since $\Delta KO_{83} = 0$, the $\mathbf{FI}_{83}$ function has an input difference $a||X$. We assume the output difference for $\mathbf{FI}_{83}$ is Y. Then, the $\mathbf{FO}_8$ function has an output difference $(a||X)||(Y \oplus (a||X))$, so the $\mathbf{FL}_9$ function has an input difference $(a||X)||(Y \oplus (a||X))$, but its output difference is indeterminate (denoted by the question marker in Figure 3.8). The $\mathbf{FL}_{10}$ function has a zero input and output difference. The same results hold for the propagation of δ under (K_B, K_D). Note that X and Y under this case may take a different value from that case under (K_A, K_C).

Finally, since the $\mathbf{FI}_{82}$ function has a zero input and output difference, by the structure of the **FO** function, we observe that only the subkeys $(K_1, K_3', K_5, K_5', K_5'^*, K_7, K_7', K_8)$ are required to check if a candidate quartet consisting of two ciphertext pairs produces the output difference $\delta = 0$ of the 7-round distinguisher. As $K_5' = \mathbf{FI}(K_5, K_6)$, $K_5'^* = K_5' \oplus b$ and $K_7' = \mathbf{FI}(K_7, K_8)$, we have the following result.

Proposition 3.4. *Only the subkeys $(K_1, K_3', K_5, K_6, K_7, K_8)$ are required to check if a candidate quartet consisting of two ciphertext pairs satisfies the output difference $\delta = 0$ of the 7-round distinguisher.*

3.4.4 (b) Attack Procedure

First, we precompute two hash tables $\mathscr{T}_1$ and $\mathscr{T}_2$ as follows.

Table $\mathscr{T}_1$. Note that $KI_{81} = K_5'$ or $K_5'^* (= K_5' \oplus b)$, $KO_{83} = K_7$ and $KI_{83} = K_3'$. Under every possible (K_3', K_5', K_7), we compute $(\Delta\mu, \Delta\nu)$ for every $x = (x_L||x_R) \in \{0,1\}^{32}$, as follows:

- $\mu = \mathbf{FI}_{81}(x_L, K_5') \oplus \mathbf{FI}_{81}(x_L, K_5' \oplus b)$,

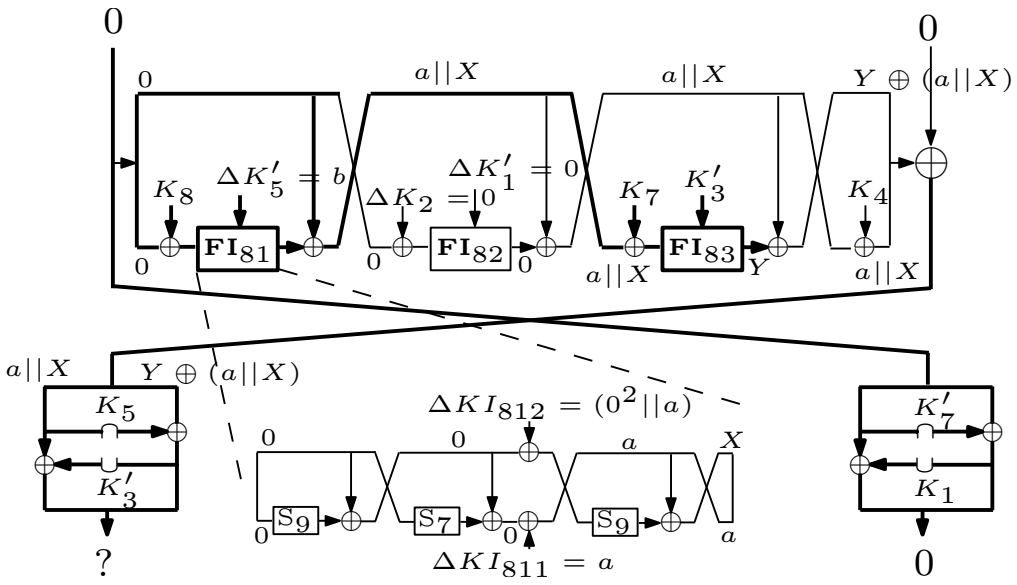


Figure 3.8: Propagation of δ Through Round 8 with $\mathbf{FL}_9$ and $\mathbf{FL}_{10}$

- $v = \mathbf{FI}_{83}(\mathbf{FI}_{81}(x_L, K'_5) \oplus x_R \oplus K_7, K'_3) \oplus \mathbf{FI}_{83}(\mathbf{FI}_{81}(x_L, K'_5 \oplus b) \oplus x_R \oplus K_7, K'_3)$.

By the structure of **FI**, we know the left 7 bits of μ must be a and μ has the form $a||X$, that is, $\mu = (a||X)$ where $X \in \mathscr{S}_a$ and $\mathscr{S}_a$ is defined above. For a fixed $(K'_3, K'_5, K_7, \mu, v)$, on average there are $2^{32} \times 2^{-8} \times 2^{-16} = 2^8$ satisfying values of x. We store the satisfying values of x into table $\mathscr{T}_1$ indexed by the value (K'_3, K'_5, K_7, X, v). There are 2^{16} possible values of K'_3, at most 2^{16} possible values of K'_5, 2^{13} possible values of K_7, 2^8 possible values of μ and 2^{16} possible values of v. Thus this precomputation takes about $2^{16} \times 2^{16} \times 2^{13} \times 2^8 \times 2^{16} \times 4 = 2^{71}$ **FI** computations and $\mathscr{T}_1$ requires a memory of about $2^{16} \times 2^{16} \times 2^{13} \times 2^8 \times 2^{16} \times 2^8 \times 4 = 2^{79}$ bytes.

Table $\mathscr{T}_2$. For every possible (K_1, K'_7, K_8), we compute $\lambda = (K_8||0^{16}) \oplus \mathbf{FL}_{10}^{-1}(x, K'_7||K_1)$ for each $x \in \{0,1\}^{32}$. There are 2^{16} possible values of K_1, 2^{13} possible values of K_7, 2^{15} possible values of K_8 and 2^{16} possible values of K'_7. Note that $K_7 = \mathbf{FI}^{-1}(K'_7, K_8)$. For a fixed (x, λ, K_7), on average there are $2^{16} \times 2^{15} \times 2^{-32} = 0.5$ satisfying values of (K_1, K'_7, K_8); for a fixed (K_1, K_7, K_8), there are 2^{32} satisfying (x, λ). We make table $\mathscr{T}_2$ in the following manner:

For every possible K_7:

For every possible (K_1, K_8):

- Compute $K_7' = \mathbf{FI}(K_7, K_8)$.
- Find all the 2^{32} possible (x, λ) such that $\lambda = (K_8||0^{16}) \oplus \mathbf{FL}_{10}^{-1}(x, K_7'||K_1)$.
- Store (K_1, K_8) into Table $\mathscr{T}_2$ indexed first by K_7 and then by (x, λ).
- Set a binary marker with two possible scenarios ("up" and "down") to the set of 2^{32} tuples $(K_7, K_1, K_8, x, \lambda)$. The marker's initial scenario is "down".

Thus, for each K_7, there are 2^{31} markers corresponding to the 2^{31} possible values of (K_1, K_8) and 2^{32} different (x, λ) that work under the same (K_7, K_1, K_8) share the same marker. $\mathscr{T}_2$ requires a memory of about $2^{13} \times 2^{16} \times 2^{15} \times 2^{32} \times 4 = 2^{78}$ bytes. This precomputation has a time complexity of about $2^{13} \times 2^{16} \times 2^{15} \times 2^{32} = 2^{76}$ $\mathbf{FL}^{-1}$ computations.

Now, we can present the following attack procedure to break the full MISTY1.

1) Initialise zero to an array of 2^{75} counters corresponding to all the 2^{75} possible values of $(K_1, K_3', K_5, K_6, K_7, K_8)$.

2) Choose a set of $2^{58.5}$ plaintext pairs $(P, P^* = P \oplus (0^{48}||b))$ and another set of $2^{58.5}$ plaintext pairs $(P', P'^* = P' \oplus (0^{48}||b))$. In a chosen plaintext attack scenario, obtain the ciphertexts for the plaintexts P, P^*, P', P'^* under K_A, K_B, K_C, K_D respectively. We denote $C = (CL_L||CL_R, CR_L||CR_R)$ as the ciphertext for plaintext P encrypted under K_A, $C^* = (CL_L^*||CL_R^*, CR_L^*||CR_R^*)$ as the ciphertext for plaintext P^* encrypted under K_B, $C' = (CL_L'||CL_R', CR_L'||CR_R')$ as the ciphertext for plaintext P' encrypted under K_C and $C'^* = (CL_L'^*||CL_R'^*, CR_L'^*||CR_R'^*)$ as the ciphertext for plaintext P'^* encrypted under K_D.

3) Check whether a candidate quartet (C, C^*, C', C'^*) meets both the following conditions by storing the ciphertext pairs (C, C^*) and (C', C'^*) into a hash table in-

dexed by the values $CR_L||CR_R||CR^*_L||CR^*_R$ and $CR'_L||CR'_R||CR'^*_L||CR'^*_R$.

$$(CR_L||CR_R) \oplus (CR'_L||CR'_R) = 0, (CR^*_L||CR^*_R) \oplus (CR'^*_L||CR'^*_R) = 0.$$

Keep only the satisfying quartets.

4) For every remaining quartet (C, C^*, C', C'^*), do the following sub-steps.

   a) Choose all the possible K'_3 satisfying the following conditions:

      - $(CL_R \cup K'_3) \oplus CL_L \oplus (CL'_R \cup K'_3) \oplus CL'_L = a||X'$,
      - $(CL^*_R \cup K'_3) \oplus CL^*_L \oplus (CL'^*_R \cup K'_3) \oplus CL'^*_L = a||X^*$,

      where X' and X^* represents two indeterminate 9-bit values (note that X', X^* can be different for different quartets, but their values are fixed for a given quartet and K'_3).

   b) For every satisfying K'_3, do as follows.

      (i) Guess K_5 and compute the difference just before the $\mathbf{FL}_9^{-1}$ function between C and C' and the difference just before the $\mathbf{FL}_9^{-1}$ function between C^* and C'^*. Let

      - $\mathbf{FL}_9^{-1}(CL_L||CL_R, K_5||K'_3) \oplus \mathbf{FL}_9^{-1}(CL'_L||CL'_R, K_5||K'_3) = a||X'||(Y' \oplus (a||X'))$,
      - $\mathbf{FL}_9^{-1}(CL^*_L||CL^*_R, K_5||K'_3) \oplus \mathbf{FL}_9^{-1}(CL'^*_L||CL'^*_R, K_5||K'_3) = a||X^*||(Y^* \oplus (a||X^*))$,

      where Y' and Y^* represent specific 16-bit values.

      (ii) Guess K_7; by Proposition 3.3-(2), it follows that there are two corresponding values of K_6 (for each guessed K_7) and we denote them by $\widetilde{K}_6$ and $\overline{K}_6$. Then, do the following four sub-steps.

      A) Compute $\widetilde{K}'_5 = \mathbf{FI}(K_5, \widetilde{K}_6); \overline{K}'_5 = \mathbf{FI}(K_5, \overline{K}_6)$.

      B) For (C, C'), access Table $\mathscr{T}_1$ at entry $(K'_3, \widetilde{K}'_5, K_7, X', Y')$ to get the possible 32-bit inputs to the $\mathbf{FO}_8$ function excluding the XOR operation with KO_{81}. As discussed earlier, when $X' \in \mathscr{S}_a$, there are 2^8 possible inputs on average and we denote them by $\widetilde{x}_1, \widetilde{x}_2, \cdots$,

$\widetilde{x}_{256}$. When X' does not belong to $\mathscr{S}_a$, there is no input and we execute Step 4(b)(ii)(D). Similarly, for (C^*, C'^*), access Table $\mathscr{T}_1$ at entry $(K'_3, \widetilde{K}'_5, K_7, X^*, Y^*)$ to get the possible 32-bit inputs to the $\mathbf{FO}_8$ function excluding the XOR operation with KO_{81}. We denote them by $\widetilde{x}^*_1, \widetilde{x}^*_2, \cdots, \widetilde{x}^*_{256}$ when $X^* \in \mathscr{S}_a$. When X' does not belong to $\mathscr{S}_a$, there is no input and we execute Step 4(b)(ii)(D).

C) For $i = 1, 2, \cdots, 256$, access Table $\mathscr{T}_2$ at entry $(K_7, CL_L || \ CL_R, \widetilde{x}_i)$ and flip the corresponding marker to "up". For $i = 1, 2, \cdots, 256$, access Table $\mathscr{T}_2$ at entry $(K_7, CL^*_L || CL^*_R, \widetilde{x}^*_i)$ and check whether the corresponding marker is "up" or "down"; if it is "up", get the corresponding (K_1, K_8) and increase 1 to the counter corresponding to the guessed $(K_1, K'_3, K_5, \widetilde{K}_6, K_7, K_8)$, or otherwise execute the next iteration (Initialise the markers in $\mathscr{T}_2$ to be "down" after finishing all the 256 iterations).

D) Repeat the above two sub-steps (B) and (C) similarly for the case $\overline{K}'_5$. When X' or X^* does not belong to $\mathscr{S}_a$, there is no input and we execute Step 4(b)(ii) with another guess for K_7. (If this sub-step is done, go to Step 4(b)(ii) and so on)

5) For a value of $(K_1, K'_3, K_5, K_6, K_7, K_8)$ whose counter has a non-zero number, exhaustively search the remaining key bits with two known plaintext-ciphertext pairs. If a value of $(K_1, K_2, \cdots, K_8)$ is suggested, output it as the secret key of the full MISTY1.

Note that in Step 4(b)(ii), we check the two pairs from a candidate quartet one after the other, instead of checking them simultaneously. This is known as the early abort technique for the (related-key) rectangle attack (Lu & Kim, 2008).

3.4.4 (c) Attack Complexity

The attack requires $2^{58.5} \times 4 = 2^{60.5}$ chosen plaintexts. There are a total of $2^{58.5} \times 2^{58.5} = 2^{117}$ candidate quartets (C, C^*, C', C'^*), of which only $2^{117} \times (2^{-32})^2 =$

2^{53} quartets are expected to satisfy the two conditions in Step 3. It takes about $2^{59.5}$ memory accesses to obtain the satisfying quartets. For every remaining quartet, there exist, on average, $2^{16} \times (2^{-7})^2 = 2^2$ possible values of K_3' satisfying the two conditions in Step 4(a). Step 4(a) has a time complexity of about $2^{53} \times 2^{16} \times 4 \times \frac{1}{2} = 2^{70}$ **FL** computations. There are a total of 2^{14} possible values of K_5, thus Step 4(b)(i) has a time complexity of $2^{53} \times 2^2 \times 2^{14} \times 4 \times \frac{1}{2} = 2^{70}$ **FL** computations (Note that some required intermediate values have been computed in Step 4(a)). On the other hand, there are a total of 2^{13} possible values of K_7, so Step 4(b)(ii)(A) has a time complexity of $2^{53} \times 2^2 \times 2^{14} \times 2^{13} \times 2 = 2^{83}$ **FI** computations. Step 4(b)(ii)(B) has a time complexity of about $2^{53} \times 2^2 \times 2^{14} \times 2^{13} \times \frac{256 \times 32}{64} + 2^{53} \times 2^2 \times 2^{14} \times 2^{13} \times 2^{-1} \times \frac{256 \times 32}{64} = 3 \cdot 2^{88}$ memory accesses (if conducted on a 64-bit computer) owing to one-bit filtering condition on X'. Due to one-bit filtering condition on X^*, Step 4(b)(ii)(C) has a time complexity of about $2^{53} \times 2^2 \times 2^{14} \times 2^{13} \times 2^{-2} \times 256 \times 2 = 2^{89}$ memory accesses. Step 4(b)(ii)(D) has a time complexity of about $3 \cdot 2^{88} + 2^{89} = 5 \cdot 2^{88}$ memory accesses.

The probability that the counter for a wrong $(K_1, K_3', K_5, K_6, K_7, K_8)$ has a non-zero number is approximately $\sum_{i=1}^{2^{117}} [\binom{2^{117}}{i} \cdot (2^{-128})^i \cdot (1-2^{-128})^{2^{117}-i}] \approx 2^{-11}$. Thus, it is expected that there are a total of $2^{75} \times 2^{-11} = 2^{64}$ wrong values of $(K_1, K_3', K_5, K_6, K_7, K_8)$ whose counters are non-zero, so in total we need to access the array of counters only 2^{64} times in Steps 4(b)(ii)(C) and 4(b)(ii)(D). The 2^{64} wrong values of $(K_1, K_3', K_5, K_6, K_7, K_8)$ make at most 2^{79} possible values of $(K_1, K_2, \cdots, K_8)$ and thus it requires $2^{79} + 2^{79} \times 2^{-64} \approx 2^{79}$ trial encryptions to check them in Step 5. In Step 5, a wrong value of $(K_1, K_2, \cdots, K_8)$ is suggested with probability $2^{-64 \times 2} = 2^{-128}$, so it is expected that there remain $2^{79} \times 2^{-128} = 2^{-49}$ values of $(K_1, K_2, \cdots, K_8)$. Thus, the number of suggested wrong user keys is rather low. Overall, the time complexity of the attack is dominated by Steps 4(b)(ii)(B), 4(b)(ii)(C) and 4(b)(ii)(D), which is $3 \cdot 2^{88} + 2^{89} + 5 \cdot 2^{88} \approx 2^{91.33}$ memory accesses plus Step 5. Therefore, by the extremely conservative estimate used in Section 3.3.3, the attack has a total time complexity of about $\frac{2^{91.33}}{16} + 2^{79} \approx 2^{87.33}$ MISTY1 encryptions.

The counter for the correct key has an expected number of $2^{117} \times 2^{-116} = 2$ and the probability that the counter for the correct key has a non-zero number is approximately $\sum_{i=1}^{2^{117}} [\binom{2^{117}}{i} \cdot (2^{-116})^i \cdot (1-2^{-116})^{2^{117}-i}] \approx 0.86$. Therefore, the related-key amplified boomerang attack has a success probability of 86%.

The memory complexity of the attack is dominated by the space for the array of 2^{75} counters, which is $2^{75} \times \frac{75}{8} \approx 2^{78.23}$ bytes. Taking the storage space for $\mathscr{T}_1$ and $\mathscr{T}_2$ into consideration, we need a total memory space of $2^{79} + 2^{78} + 2^{78.23} \approx 2^{80.07}$ bytes.

It is very worthy to note that we can slightly reduce the memory space by splitting $\mathscr{T}_1$ into two smaller tables which mainly correspond to $\mathbf{FI}_{81}$ and $\mathbf{FI}_{83}$ respectively, but at the cost of a few more memory accesses in the attack procedure.

3.4.5 Three Other Classes of 2^{90} Weak Keys

The above sub-sections have shown a class of 2^{90} weak keys and a related-key amplified boomerang attack on the full MISTY1 under a weak key. Nevertheless, there exist three other classes of 2^{90} weak keys under which there are similar results. The new weak key classes are obtained by setting other possible values of the two subkey bits $(K_{5,3}, K_{5,12})$, which are further classified into several sub-classes by the possible values of the two subkey bits combination $(K'_{3,3}, K'_{3,12})$. This will affect only the $\mathbf{FL}_2$ function of the first related-key differential and the input difference of $\mathbf{FL}_2$ will be fixed once the setting is given, provided that the output difference of $\mathbf{FL}_2$ is $0^9||a||b$. Likewise, by choosing a number of plaintext pairs with a corresponding difference, we can conduct a similar attack on the full MISTY1 under every sub-class of weak keys. In total, we have 2^{92} weak keys under which a related-key amplified boomerang attack can break the full MISTY1.

One might consider obtaining more weak keys by setting $K'_{4,3} = 1$, instead of $K'_{4,3} = 0$ used in our results. This case will affect only the output difference of the $\mathbf{FL}_4$ function of the first related-key differential and it seems that we can further classify the resulting class of weak keys into two sub-classes according to the possible values

of the subkey bit $K_{6,3}$ as we did before. However, this case is not possible because $K_{6,3} \oplus K^*_{6,3} = 1$ and a detailed analysis reveals that under the condition that the input difference of $\mathbf{FL}_4$ is $c||0^{16}$, the output difference of $\mathbf{FL}_4$ under one plaintext pair from a candidate quartet is definitely not equal to the output difference of $\mathbf{FL}_4$ under the other plaintext pair from the candidate quartet. Consequently, the XOR of the four differences concerned between the two sub-ciphers when constructing an amplified boomerang distinguisher is non-zero, so the four related-key differentials cannot form an amplified boomerang distinguisher.

3.5 Summary

The MISTY1 block cipher has received considerable attention and its security has been thoroughly analysed since its publication. In particular, the European NESSIE project announced that "no weaknesses were found in the selected designs" when making the portfolio of selected cryptographic algorithms including MISTY1. In this chapter, we have described $2^{103.57}$ weak keys for a related-key differential attack on the full MISTY1 and 2^{92} weak keys for a related-key amplified boomerang attack on the full MISTY1.

For the very first time, our results exhibit a cryptographic weakness in the full MISTY1 cipher algorithm, particularly from an academic point of view where the cipher does not behave like an ideal cipher (in the related-key model) and thus it cannot be regarded as an ideal cipher. From a practical point of view, our attacks do not pose a significant threat to the security of MISTY1, for they work under the assumptions of weak-key and related-key scenarios and their complexity is beyond current computational capabilities. Nonetheless, the weak key classes mean that a large fraction of all possible 2^{128} keys in the whole key space of MISTY1 is weak in the sense of related-key cryptanalysis, that is, roughly one of every twenty-two million keys in the larger set of $2^{103.57}$ weak keys and thus the chance of picking such a weak key at random is not trivial. In this sense, the presence of these weak keys has an impact on the security of the full MISTY1 cipher.

CHAPTER 4

DIFFERENTIAL ATTACK ON NINE ROUNDS OF THE SEED BLOCK CIPHER

The SEED block cipher has a 128-bit block length, a 128-bit secret key and is composed of a total of 16 rounds. It is an ISO standard. In this chapter, we describe two 7-round differentials with probabilities that are slightly greater than the best previously known one on SEED. With these differentials, we present a differential cryptanalysis attack on a 9-round reduced version of SEED. The attack requires a memory of $2^{69.71}$ bytes and has a time complexity of $2^{126.36}$ encryptions with a success probability of 99.9% given 2^{125} chosen plaintexts, or a time complexity of $2^{125.36}$ encryptions with a success probability of 97.8% given 2^{124} chosen plaintexts. Our result is better than any previously published cryptanalytic results on SEED in terms of the numbers of attacked rounds and it suggests for the first time that the safety margin of SEED decreases below half of the number of rounds.

4.1 Introduction

The SEED block cipher (KISA, 1998) was designed by a group of Korean cryptographers in 1998. It has a 128-bit block length, a 128-bit secret key and a total number of 16 rounds. SEED became a Korean national industrial association standard (TTA, 1999) and was adopted as an ISO standard (ISO, 2005, 2010). Currently, SEED is deployed in real applications, particularly by banks and companies in Korea, to protect the privacy of the users and the transaction data in security applications like e-commerce and financial services. Further, it was included in PKCS #11 on Cryptographic Token Interface Standard (PKCS, 2009) and was proposed by Internet Engineering Task Force (IETF) for Cryptographic Message Syntax (CMS) (Park, Lee, Kim, & Lee, 2005), Transport Layer Security (TLS) (H. Lee, Yoon, & Lee, 2005), Secure Real-time Transport Protocol (SRTP) (Yoon, Kim, Park, Jeong, & Won, 2010)

and IPsec (H. Lee, Yoon, Lee, & Lee, 2005). In addition, Mozilla Firefox web browser supports the SEED algorithm now.

The designers of SEED analysed its security against differential cryptanalysis (Biham & Shamir, 1991b) as well as certain other cryptanalytic techniques, claiming that a 6-round differential characteristic of SEED would have a probability of at least 2^{-130}, implying that an effective 6-round differential characteristic for SEED will not exist. However, Yanami and Shimoyama (2003) presented three 6-round differential characteristics with probability 2^{-124} and finally used them to conduct a differential attack on 7-round SEED with a time complexity faster than exhaustive key search. Sung (2011) described a 7-round differential with probability 2^{-122} for SEED, by summing the probabilities of many 7-round differential characteristics with the same input and output differences and finally gave a differential attack on 8-round SEED; Sung also described a 7-round differential with probability 2^{-124} of SEED. Sung's attack on 8-round SEED is the best previously published cryptanalytic result on the SEED cipher algorithm in terms of the number of attacked rounds.

In this chapter, we further investigate the security of SEED against differential cryptanalysis. Specifically, we present two 7-round differentials with probabilities that are slightly larger than the probability of Sung's best 7-round differential, plus seventeen 7-round differentials with probabilities which are slightly larger than the probability of Sung's second best 7-round differential. More importantly, we devise a differential attack on 9-round SEED, which requires a memory of $2^{69.71}$ bytes and has a time complexity of $2^{126.36}$ encryptions with a success probability of 99.9% given 2^{125} chosen plaintexts, or a time complexity of $2^{125.36}$ encryptions with a success probability of 97.8% given 2^{124} chosen plaintexts. This is the first published cryptanalytic attack on 9-round SEED and it suggests that the safety margin of SEED decreases below half of the number of rounds. Table 4.1 summarises previous results and our main cryptanalytic results on SEED.

Table 4.1: Main Cryptanalytic Results of Differential Cryptanalysis on SEED

Rounds	Data	Memory	Time	Success Prob.	Source
7	2^{126}	2^{67}	$2^{126.37}$	68.8%	(Yanami & Shimoyama, 2003)
8	2^{125}	2^{67}	$2^{125.17}$	98.1%	(Sung, 2011)
9	2^{125}	$2^{69.71}$	$2^{126.36}$	99.9%	Sect. 4.4
9	2^{124}	$2^{69.71}$	$2^{125.36}$	97.8%	Sect. 4.4

Data unit: Chosen plaintexts, Memory unit: Bytes, Time unit: Encryptions.

Organisation. The remainder of this chapter is organised as follows. In the next section, we describe the SEED block cipher. In Section 4.3, we describe the 7-round differentials of SEED. In Section 4.4, we present our differential attack on 9-round SEED. Section 4.5 concludes the chapter.

4.2 The SEED Block Cipher

SEED (KISA, 1998) employs a typical Feistel structure shown in Figure 4.1 with a 128-bit block size and a 64-bit round subkey. However, the round function **F** of SEED is very complex, building on the **G** function. Below we describe the **G** and **F** functions in greater detail.

- **The G function.** There are two layers in the **G** function, see Figure 4.2. The first layer involves two 8×8-bit S-boxes S_1 and S_2 which are constructed from two different Boolean functions. The second layer is a permutation which processes the output of the first layer, made up of a bitwise AND operation with four specific values c_1, c_2, c_3 and c_4 (see (KISA, 1998)), followed by an XOR operation on the expanded 16 blocks. Given a four-byte input (X_4, X_3, X_2, X_1), the **G** function generates a four-byte output (Y_4, Y_3, Y_2, Y_1) as follows:
  - $Y_1 = (S_1(X_1)\&c_1) \oplus (S_2(X_2)\&c_2) \oplus (S_1(X_3)\&c_3) \oplus (S_2(X_4)\&c_4)$,
  - $Y_2 = (S_1(X_1)\&c_2) \oplus (S_2(X_2)\&c_3) \oplus (S_1(X_3)\&c_4) \oplus (S_2(X_4)\&c_1)$,
  - $Y_3 = (S_1(X_1)\&c_3) \oplus (S_2(X_2)\&c_4) \oplus (S_1(X_3)\&c_1) \oplus (S_2(X_4)\&c_2)$,
  - $Y_4 = (S_1(X_1)\&c_4) \oplus (S_2(X_2)\&c_1) \oplus (S_1(X_3)\&c_2) \oplus (S_2(X_4)\&c_3)$.

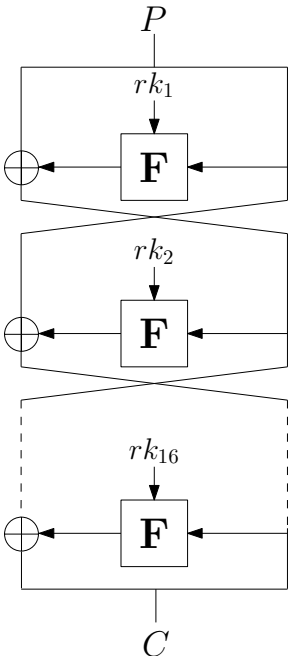


Figure 4.1: The Feistel Structure of the SEED Block Cipher

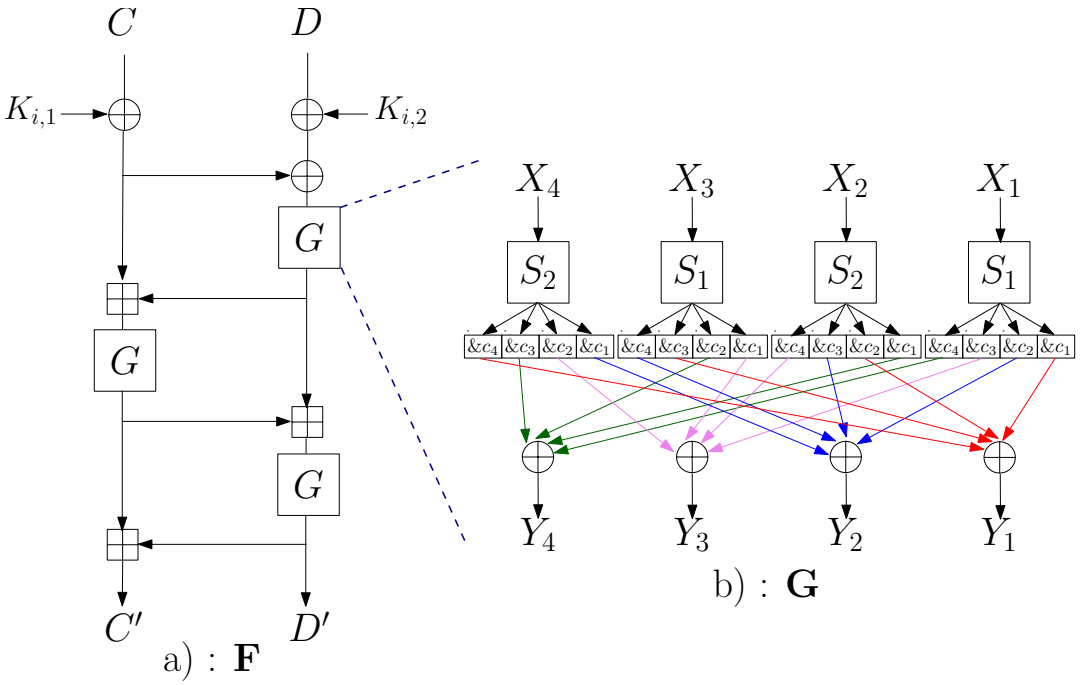


Figure 4.2: The Structures of the F and G Functions

- **The F function.** As depicted in Figure 4.2, a 64-bit input block of the **F** function is divided into two 32-bit blocks *C* and *D*. After being XORed with the two 32-

bit subkeys $K_{i,1}$ and $K_{i,2}$ respectively, the two blocks pass through three layers of **G** function and finally the two 32-bit output blocks C' and D' of the **F** function are computed as follows:

- $C' = \mathbf{G}(\mathbf{G}(\mathbf{G}((C \oplus K_{i,1}) \oplus (D \oplus K_{i,2})) \boxplus (C \oplus K_{i,1})) \boxplus \mathbf{G}((C \oplus K_{i,1}) \oplus (D \oplus K_{i,2}))) \boxplus \mathbf{G}(\mathbf{G}((C \oplus K_{i,1}) \oplus (D \oplus K_{i,2})) \boxplus (C \oplus K_{i,1}))$,
- $D' = \mathbf{G}(\mathbf{G}(\mathbf{G}((C \oplus K_{i,1}) \oplus (D \oplus K_{i,2})) \boxplus (C \oplus K_{i,1})) \boxplus \mathbf{G}((C \oplus K_{i,1}) \oplus (D \oplus K_{i,2})))$.

SEED uses a total of sixteen 64-bit subkeys $rk_i = K_i$ for the round functions $i = 1, 2, \cdots, 16$. All subkeys are derived from a 128-bit secret key K and each round subkey K_i is made up of two 32-bit subkeys $K_i = (K_{i,1}, K_{i,2})$. The `Key Schedule` algorithm works as follows:

1) Represent K as four 32-bit words $K = (K_d, K_c, K_b, K_a)$.

2) The subkeys are generated as follows where $\widehat{c_i}$ are specific constants (see (KISA, 1998)). For $i = 1, 2, \cdots, 16$, compute:

   - $K_{i,1} = \mathbf{G}(K_b \boxplus K_d \boxminus \widehat{c_i})$.
   - $K_{i,2} = \mathbf{G}(K_c \boxminus K_a \boxplus \widehat{c_i})$.
   - If $i \bmod 2 = 1$, then $(K_d || K_c) = (K_d || K_c) \ggg 8$; else $(K_b || K_a) = (K_b || K_a) \lll 8$.

SEED takes a 128-bit plaintext as input and the encryption algorithm `E` works as follows. A 128-bit plaintext P is divided into two 64-bit blocks $P = (P_L, P_R)$. The right 64-bit block P_R is input to the **F** function with the 64-bit round subkey K_1. The output of the **F** function is XORed with the left 64-bit block P_L which becomes the right 64-bit input block to the second round while P_R becomes the left 64-bit input block to the second round. After 16 similar encryption rounds, the final 128-bit output is the ciphertext of the plaintext.

4.3 Seven-Round Differentials of SEED

In this section, we first describe the 7-round differentials owing to Sung (2011) and then present two 7-round differentials with probabilities which are slightly larger than Sung's best 7-round differential and seventeen 7-round differentials with probabilities slightly larger than Sung's second best 7-round differential.

4.3.1 Sung's 7-Round Differentials

Sung (2011) presented a 7-round differential $(0x80808000, 0, 0x87808000, 0x80808000) \xrightarrow{7} (0x07808000, 0x80808000, 0x80808000, 0)$ with probability 2^{-122} against SEED and a 7-round differential $(0x80808000, 0, 0x83808000, 0x80808000) \xrightarrow{7} (0x03808000, 0x80808000, 0x80808000, 0)$ with probability 2^{-124}, which were obtained by summing the probabilities of many 7-round differential characteristics with the same input and output differences. Refer to Sung (2011) for an illustration of the 7-round differentials. Otherwise, see Figure 4.3-(a) where $\alpha = 0x80808000$ and $X = 0x87808000$ or $0x83808000$.

It is worthy to mention that our computation shows that the probabilities 2^{-122} and 2^{-124} for Sung's 7-round differentials are more precise about $2^{-121.21}$ and $2^{-122.84}$ respectively if we keep two more significant digits. We will use these more accurate values in our subsequent analysis.

4.3.2 Our 7-Round Differentials

We performed computer simulations to search 7-round differentials of SEED over sixteen different differential patterns by considering many 7-round differential characteristics with the same input and output differences. We find some valuable results in two patterns that are depicted in Figure 4.3-(a) and (b) as follows.

- Sung's 7-round differential with probability $2^{-121.21}$ is one of the best 7-round differentials (i.e., 7-round differentials with the largest probability) in Pattern (I).

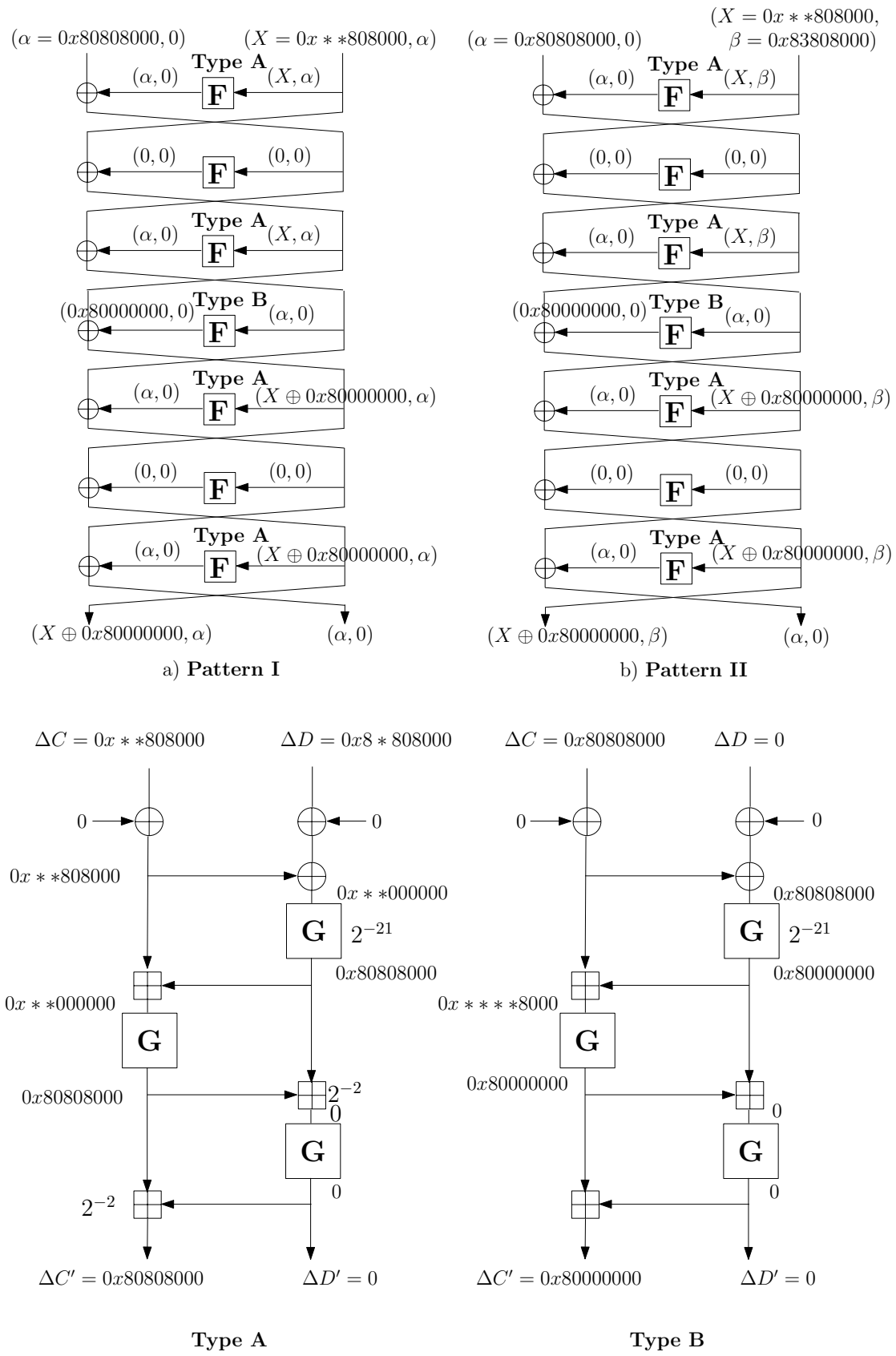


Figure 4.3: 7-Round Differentials of SEED

In Pattern (I), there is another 7-round differential with probability $2^{-121.21}$ and eight additional 7-round differentials with a probability ranging from $2^{-121.22}$ to $2^{-122.81}$, which are larger than the probability of Sung's second best 7-round differential (with probability of $2^{-122.84}$).

- There are two 7-round differentials with probability $2^{-121.07}$ in Pattern (II) which are slightly larger than the probability of Sung's best 7-round differential (with the probability of $2^{-121.21}$). Besides, there are eight additional 7-round differentials with a probability ranging from $2^{-121.22}$ to $2^{-122.81}$, which are larger than the probability of Sung's second best 7-round differential (with probability of $2^{-122.84}$).

We summarise all these new 7-round differentials in Table 4.2. As an example, here we give the probabilities for the rounds of one of our best 7-round differentials, i.e., $(0x80808000, 0, 0x84808000, 0x83808000) \xrightarrow{7} (0x04808000, 0x83808000, 0x80808000, 0)$, which is the first with Pattern (II) given in Table 4.2. This 7-round differential is also the one that we will use in Section 4.4. Following Figure 4.3-(b), we deduce that the first and third rounds have a probability of approximately $2^{-18.45}$ each, the fourth round has a probability of approximately $2^{-43.79}$ and the fifth and seventh rounds have a probability of approximately $2^{-20.19}$ each. Clearly, the second and sixth rounds have a probability of one. Hence, the 7-round differential has a total probability of $2^{-18.45\times 2} \times 2^{-43.79} \times 2^{-20.19\times 2} = 2^{-121.07}$.

It is interesting to see from Table 4.2 that the differentials appear pairwise with the same probability due to the symmetry of the Feistel structure for forward and backward directions. Our best 7-round differential in Pattern (I) is the counterpart of Sung's best 7-round differential. We would like to mention that there exist a large number of 7-round differentials whose probabilities are not as large as the probability of $2^{-122.84}$ of Sung's second best 7-round differential but is still bigger than 2^{-128}.

Table 4.2: Our 7-Round Differentials of SEED

Pattern	X	$X \oplus 0x8000000$	Probability
(I)	$0x07808000$	$0x87808000$	$2^{-121.21}$
	$0x87808000$	$0x07808000$	$2^{-121.21}$
	$0x44808000$	$0xC4808000$	$2^{-121.22}$
	$0xC4808000$	$0x44808000$	$2^{-121.22}$
	$0x45808000$	$0xC5808000$	$2^{-121.96}$
	$0xC5808000$	$0x45808000$	$2^{-121.96}$
	$0x4C808000$	$0xCC808000$	$2^{-122.55}$
	$0xCC808000$	$0x4C808000$	$2^{-122.55}$
	$0x47808000$	$0xC7808000$	$2^{-122.81}$
	$0xC7808000$	$0x47808000$	$2^{-122.81}$
(II)	$0x84808000$	$0x04808000$	$2^{-121.07}$
	$0x04808000$	$0x84808000$	$2^{-121.07}$
	$0x44808000$	$0xC4808000$	$2^{-121.22}$
	$0xC4808000$	$0x44808000$	$2^{-121.22}$
	$0x45808000$	$0xC5808000$	$2^{-121.96}$
	$0xC5808000$	$0x45808000$	$2^{-121.96}$
	$0x00808000$	$0x80808000$	$2^{-122.54}$
	$0x80808000$	$0x00808000$	$2^{-122.54}$
	$0x47808000$	$0xC7808000$	$2^{-122.81}$
	$0xC7808000$	$0x47808000$	$2^{-122.81}$

4.4 Differential Attack on 9-Round SEED

In this section, we devise a differential attack on 9-round SEED based on the best 7-round differential (that has a probability of $2^{-121.07}$) described in Section 4.3.2 with $\alpha = 0x80808000, \beta = 0x83808000$ and $X = 0x84808000$. Thus, $\widehat{X} = X \oplus 0x80000000 = 0x04808000$. The attack consists of an offline precomputation phase and an online attack phase. Without loss of generality, we assume the attacked rounds are the first 9 rounds of SEED, that is, from Rounds 1 to 9. It is noteworthy that similar cryptanalytic results can be obtained by using certain other 7-round differentials including Sung's 7-round differentials.

4.4.1 Precomputation

Suppose x is the 64-bit state immediately after the XOR operations with the round subkey K_9 in the **F** function of Round 9. We precompute a table $\mathscr{T}_x$, so that, given an output difference of the **F** function, we only need a single table lookup (memory access) to $\mathscr{T}_x$ to retrieve the pair of values at state x that have the difference $(\alpha, 0)$ and generate the given output difference after the **F** function. We generate the table $\mathscr{T}_x$ as follows:

- For every possible 64-bit value x such that $x < [x \oplus (\alpha, 0)]$:
  - Compute the output difference immediately after the **F** function corresponding to the pair of values $(x, x \oplus (\alpha, 0))$ and denote it by Δy.
  - Store x into Table $\mathscr{T}_x$ indexed by y.

There are 2^{63} possible values for x and at most 2^{63} possible values for y. Thus, the precomputation requires a memory of $2^{63} \times \frac{64}{8} = 2^{66}$ bytes and has a time complexity of about $2^{63} \times 2 = 2^{64}$ computations of the **F** function (which are approximately equivalent to $2^{64} \times \frac{1}{9} \approx 2^{60.84}$ 9-round SEED encryptions) and 2^{63} memory accesses (which are equivalent to $\frac{2^{63}}{9} \approx 2^{59.84}$ 9-round SEED encryptions by our estimate given in Section 4.4.3). There are a total of 2^{63} possible combinations of (x, y) and there is, on average, only one value of x in every entry y of table $\mathscr{T}_x$. Occasionally, there may be more than one value of x in an entry of table $\mathscr{T}_x$, but it does not affect the correctness of our attack and an average number of 1 still holds.

4.4.2 Attack Procedure

Now, we present the following attack procedure to break the first 9 rounds of SEED. Recall that $\alpha = 0x80808000, \beta = 0x83808000$, $X = 0x84808000$ and $\widehat{X} = 0x04808000$, all of which are fixed for this specific attack. The attack is illustrated in Figure 4.4.

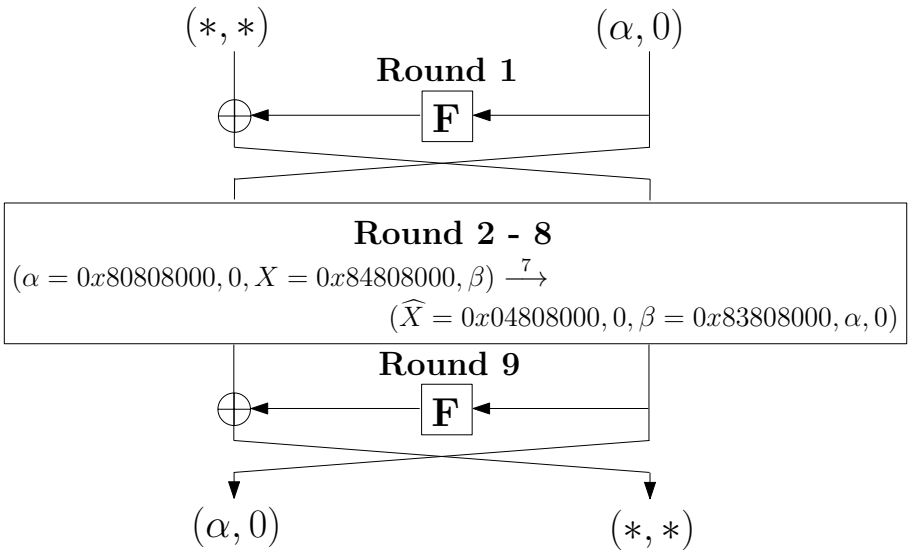


Figure 4.4: Differential Attack on 9-Round SEED

1. Choose 2^{ϕ} structures $\mathscr{S}_i$ (a specific value of ϕ will be given below and for $i = 1,2,\cdots,2^{\phi}$) where a structure is defined to be a set of 2^{64} plaintexts $P_{i,j}$ (for $j = 1,2,\cdots,2^{64}$) with the left half taking all the possible 64-bit values, bits (48, 56, 64) of the right half fixed to a certain value such that its complement is not used as bits (48, 56, 64) of the right half of any other structure and the other 61 bits fixed. In a chosen plaintext attack scenario, obtain all the ciphertexts for the 2^{64} plaintexts in each of the 2^{ϕ} structures. We denote $C_{i,j}$ as the ciphertext for plaintext $P_{i,j}$.

2. Choose 2^{ϕ} structures $\widehat{\mathscr{S}_i}$ (for $i = 1,2,\cdots,2^{\phi}$) where a structure $\widehat{\mathscr{S}_i}$ is obtained by taking the complement of bits (48, 56, 64) of the right half of all the plaintexts $P_{i,j}$ in $\mathscr{S}_i$. In a chosen plaintext attack scenario, obtain all the ciphertexts for the 2^{64} plaintexts in each $\widehat{\mathscr{S}_i}$.

3. For each pair of structures $(\mathscr{S}_i, \widehat{\mathscr{S}_i})$, perform the following three sub-steps:

   a) For $1 \le l \le 2^{64}$, identify all plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ (or more precisely, the quartets $(P^L_{i,j}, C^R_{i,j}, \widehat{P}^L_{i,l}, \widehat{C}^R_{i,l})$) such that $C^L_{i,j} \oplus \widehat{C}^L_{i,l}$ is equal to $(\alpha,0)$ using a sorting (or hash) method. For example, by storing $(P_{i,j}, C_{i,j})$ ($(P^L_{i,j}, C^R_{i,j})$ respectively) indexed by $C^L_{i,j}$ and storing $(\widehat{P}_{i,j}, \widehat{C}_{i,j})$

$((\widehat{P}^L_{i,j}, \widehat{C}^R_{i,j})$ respectively) indexed by $\widehat{C}^L_{i,j} \oplus (\alpha, 0)$, where $P^L_{i,j}$ and $\widehat{P}^L_{i,l}$ represent the left halves of $P_{i,j}$ and $\widehat{P}_{i,l}$ respectively, $C^R_{i,j}$ and $\widehat{C}^R_{i,l}$ represent the right half of $C_{i,j}$ and $\widehat{C}_{i,l}$ respectively and $C^L_{i,j}$ and $\widehat{C}^L_{i,l}$ represent the left halves of $C_{i,j}$ and $\widehat{C}_{i,l}$ respectively.

b) Store the satisfying plaintext-ciphertext quartets $(P_{i,j}, \widehat{P}_{i,l}, C_{i,j}, \widehat{C}_{i,l})$ (or more concisely, the pairs $(C^L_{i,j}, C^R_{i,j} \oplus \widehat{C}^R_{i,l})$) into a table $\mathscr{T}_P$ indexed by $P^L_{i,j} \oplus \widehat{P}^L_{i,l}$.

c) Guess a value for the round subkey K_1 and perform the following sub-steps:

(i) Partially encrypt the two right halves respectively from the plaintexts in $\mathscr{S}_i$ and $\widehat{\mathscr{S}_i}$ through the **F** function of Round 1 and denote the output difference of the **F** function as Δ.

(ii) Access entry $(\Delta \oplus (X, \beta))$ in Table $\mathscr{T}_P$ to get the plaintext-ciphertext quartet and assume it is $(P_{i,j}, \widehat{P}_{i,l}, C_{i,j}, \widehat{C}_{i,l})$ (the pair $(C^L_{i,j}, C^R_{i,j} \oplus \widehat{C}^R_{i,l})$ respectively).

(iii) Given the selected plaintext-ciphertext quartet $(P_{i,j}, \widehat{P}_{i,l}, C_{i,j}, \widehat{C}_{i,l})$ (the selected pair $(C^L_{i,j}, C^R_{i,j} \oplus \widehat{C}^R_{i,l})$ respectively), access entry $((\widehat{X}, \beta) \oplus C^R_{i,j} \oplus \widehat{C}^R_{i,l})$ in the precomputation table $\mathscr{T}_x$ to get the intermediate value(s) immediately after the XOR operations with the round subkey K_9 in the **F** function of Round 9 and denote it as z. Then, compute two possible values for K_9 as $z \oplus C^L_{i,j}$ and $z \oplus C^L_{i,j} \oplus (\alpha, 0)$. Note that $C^L_{i,j} \oplus (\alpha, 0) = \widehat{C}^L_{i,l}$.

(iv) Recover the corresponding secret key from each of the obtained candidate values on (K_1, K_9) and check whether the secret key is correct with one or more plaintext-ciphertext pairs. If it passes this test, then it is very likely to be the correct user key and we terminate the procedure; otherwise, repeat Step 3 with another pair of structures.

Notice that there are only two possible 64-bit values for the right halves of the plaintexts in a pair of structures $(\mathscr{S}_i, \widehat{\mathscr{S}_i})$ and it is simple to recover the secret key from a candidate (K_1, K_9) by the key schedule of SEED. The overall idea used in this

attack is similar to that used by Biham and Shamir (1993) to break the full DES block cipher (NBS, 1977).

4.4.3 Attack Complexity

The attack requires $2 \times 2^{\phi} \times 2^{64} = 2^{\phi+65}$ chosen plaintexts. Steps 1 and 2 have a time complexity of $2^{\phi+65}$ 9-round SEED encryptions. Observe that we collect another pair of plaintext structures after testing a pair of plaintext structures, so that we can reuse the memory for storing the pair of plaintext structures. Furthermore, for a structure $\mathscr{S}_i$ of 2^{64} plaintexts $P_{i,j}$, we store the (identical) right halves $P^R_{i,j}$ of the plaintexts only once, then store $(P^L_{i,j}, C^R_{i,j})$ indexed by $C^L_{i,j}$ and similarly for the other structure $\widehat{\mathscr{S}_i}$. Thus, the memory complexity of the online attack is dominated by the space for storing the pair of plaintext structures and table $\mathscr{T}_P$, which is approximately $2^{64} \times (8+8) \times 2 + 2^{64} \times (8+8) = 3 \times 2^{68}$ bytes.

Next, we analyse Steps 3(a)–3(c) for a pair of structures $(\mathscr{S}_i, \widehat{\mathscr{S}_i})$. Step 3(a) has a time complexity of about $2^{64} \times 2 = 2^{65}$ memory accesses. There is a 64-bit filtering condition in Step 3(a) and thus it is expected that there are $2^{64} \times 2^{64} \times 2^{-64} = 2^{64}$ satisfying ciphertext pairs $(C_{i,j}, \widehat{C}_{i,l})$ after Step 3(a). Thus, Step 3(b) has a time complexity of about 2^{64} memory accesses. Since there are only two possible 64-bit values for the right halves of the plaintexts in a pair of structures $(\mathscr{S}_i, \widehat{\mathscr{S}_i})$, Step 3(c)(i) has a time complexity of approximately $2^{64} \times 2 \times \frac{1}{9} = \frac{2^{65}}{9}$ 9-round SEED encryptions. For a guessed K_1, there is about only one memory access in Step 3(c)(ii)/3(c)(iii) and Step 3(c)(iv) has a time complexity of about $4 \times 2 = 8$ computations of the **G** function to recover the two possible user keys by the key schedule of SEED which are approximately equal to three computations of the **F** function plus 2 trial 9-round SEED encryptions. Hence, for all 2^{ϕ} pairs of structures, Step 3 has a total time complexity of approximately $2^{\phi} \times \frac{2^{65}}{9} + 2^{\phi} \times 2^{64} \times \frac{3}{9} + 2^{\phi} \times 2^{64} \times 2 = \frac{23}{9} \times 2^{\phi+64}$ 9-round SEED encryptions and $2^{\phi} \times 2^{65} + 2^{\phi} \times 2^{64} + 2^{\phi} \times 2^{64} \times (1+1) = 5 \times 2^{\phi+64}$ memory accesses.

In total, the online attack has a time complexity of approximately $2^{\phi+65} + \frac{23}{9} \times 2^{\phi+64} = \frac{41}{9} \times 2^{\phi+64}$ 9-round SEED encryptions and $5 \times 2^{\phi+64}$ memory accesses.

As mentioned in Section 3.3.3 (c), the question on how many memory accesses (table lookups) are equivalent to one SEED encryption in terms of time depends closely on the used platform and SEED implementation as well as the storage location of the sorted table. In theoretical block cipher cryptanalysis, it is usually assumed by default that a table is stored in an ideal place, RAM say, like an S-box table and it takes an almost constant time to access an entry in a sorted table independently of the number of entries. Thus, an extremely conservative estimate is that 9 memory accesses equal a 9-round SEED encryption in terms of time assuming that the **F** function without the XOR operations with a round subkey is precomputed in a table and is equivalent to one memory access by neglecting the computational complexity for other operations and the key schedule. Thus, one round is equivalent to one memory access. As a consequence, the total time complexity of the online attack is $\frac{41}{9} \times 2^{\phi+64} + \frac{5\times 2^{\phi+64}}{9} = \frac{46}{9} \times 2^{\phi+64}$ 9-round SEED encryptions.

By taking the complexity for the precomputation into consideration, the attack requires a total memory of approximately $3 \times 2^{68} + 2^{66} \approx 2^{69.71}$ bytes and a total time complexity of approximately $\frac{46}{9} \times 2^{\phi+64} \approx 2^{126.36}$ 9-round SEED encryptions, when we let $\phi = 60$.

The attack succeeds if there is at least one right plaintext pair. When $\phi = 60$, for each candidate (K_1, K_9) there are $2^{\phi} \times 2^{64} = 2^{124}$ candidate plaintext pairs that have an input difference $(\alpha, 0, \star, \star)$ to Round 2 and an output difference $(\star, \star, \alpha, 0)$ immediately after Round 8 (that is, after Step 3(a)), thus the attack has an expected success probability of approximately $1 - (1 - 2^{-121.07})^{2^{124}} \approx 1 - \mathrm{e}^{-2^{2.93}} \approx 99.9\%$.

We can obtain a faster attack with a smaller success probability by using a smaller number of data, for example, if we let $\phi = 59$, (that is, 2^{59} pairs of plaintext structures — a total of 2^{124} chosen plaintexts). Then the resulting attack requires the same amount of memory of $2^{69.71}$ bytes, but has a total time complexity of $\frac{46}{9} \times 2^{\phi+64} \approx 2^{125.36}$ 9-round SEED encryptions, with an expected success probability of approximately $1 - (1 - 2^{-121.07})^{2^{123}} \approx 1 - \mathrm{e}^{-2^{1.93}} \approx 97.8\%$.

Observe that all the table lookups are operated on either 64-bit or 128-bit data with a 64-bit index and we assume each table lookup is done with a single memory access as widely adopted in theoretical analysis. However, on a 64-bit computer in reality, it takes two memory accesses to retrieve 128-bit data with a 64-bit index, thus the resulting number of memory accesses will be $2^{\phi} \times 2^{65} \times 2 + 2^{\phi} \times 2^{64} \times 2 + 2^{\phi} \times 2^{64} \times (2+1) = 9 \times 2^{\phi+64}$ and the resulting total time complexity of the attack will be $\frac{41}{9} \times 2^{\phi+64} + \frac{9 \times 2^{\phi+64}}{9} \approx 2^{\phi+66.48}$ 9-round SEED encryptions which is still smaller than that for exhaustive key search when $\phi = 60$ or 59.

It is worthy to notice that the structure pairs $(\mathscr{S}_i, \widehat{\mathscr{S}_i})$ can be generated by using different keys, like what Biham and Shamir (1993) mentioned for their DES attack. Under this scenario, we can find one or more of the used secret keys as long as there is a right plaintext pair.

4.4.4 Notes

In this subsection, we describe two time-memory trade-offs to the above attack and compute the success probabilities of Yanami and Shimoyama's 7-round attack and Sung's 8-round attack.

4.4.4 (a) Note 1

Observe that the round functions of Rounds 1 and 9 have the same input difference $(\alpha, 0)$, thus we can obtain a time-memory trade-off by making the following revisions to the above attack. For every satisfying plaintext-ciphertext quartet after Step 3(a), with a single lookup to entry $((X, \beta) \oplus P^L_{i,j} \oplus \widehat{P}^L_{i,l})$ in table $\mathscr{T}_x$, we retrieve intermediate value(s) immediately after the XOR operations with the round subkey K_1 in the **F** function of Round 1 and we denote it by $\widehat{z}$, followed by computing the two possible values for K_1 as $K_1 = \widehat{z} \oplus P^R_{i,j}$ and $K_1 = \widehat{z} \oplus \widehat{P}^R_{i,l}$. We then retrieve the two possible values for K_9 with a single lookup to $\mathscr{T}_x$ and finally check whether one of the four combinations on (K_1, K_9) produces the correct secret key.

For a structure, we only need to store $(P^L_{i,j}, C^R_{i,j})$ indexed by $C^L_{i,j}$. As a result, this time–memory trade-off requires a total memory of approximately $2^{64} \times (8+8) \times 2 + 2^{66} \approx 2^{69.17}$ bytes. This time–memory trade-off has a total time complexity of approximately $2^{125} + \frac{2^{60} \times 2^{64} \times (1+1+1)}{9} + 2^{60} \times 2^{64} \times \frac{4 \times 4}{9 \times 3} + 2^{60} \times 2^{64} \times 4 \approx 2^{126.8}$ 9-round SEED encryptions when using 2^{60} pairs of plaintext structures with the same success probability of 99.9%. Meanwhile, it has a total time complexity of approximately $2^{124} + \frac{2^{59} \times 2^{64} \times (1+1+1)}{9} + 2^{59} \times 2^{64} \times \frac{4 \times 4}{9 \times 3} + 2^{59} \times 2^{64} \times 4 \approx 2^{125.8}$ 9-round SEED encryptions when using 2^{59} pairs of plaintext structures with the same success probability of 97.8%. Each version is slightly slower than the corresponding attack version described in the last subsection, but requires a slightly smaller memory.

4.4.4 (b) Note 2

Another time–memory trade-off can be obtained by precomputing a table so that we can retrieve the candidate subkey K_9 with a single table lookup to this table, given a pair of input blocks to the **F** function with difference $(\alpha, 0)$ and their output difference immediately after the **F** function. A straightforward way to generate such a table is as follows:

- For every possible 64-bit value u such that $u < [u \oplus (\alpha, 0)]$:
  - For every possible value of the round subkey K_9:
    - Compute $w = \mathbf{F}(u, K_9) \oplus \mathbf{F}(u \oplus (\alpha, 0), K_9)$.
    - Store K_9 into the table indexed by (u, w).

There are 2^{63} possible values for u and at most $2^{64} - 1$ possible values for w. Thus, the precomputation requires a memory of $2^{63} \times (2^{64} - 1) \times \frac{64}{8} \approx 2^{130}$ bytes (this is smaller than, more specifically, a quarter of a memory of $2^{132} (= 2^{128} \times 16)$ bytes required by the dictionary or codebook attack) and has a time complexity of about $2^{63} \times 2^{64} \times 2 = 2^{128}$ computations of the **F** function (which are approximately equivalent to $2^{128} \times \frac{1}{9} \approx 2^{124.84}$ 9-round SEED encryptions) and $2^{63} \times 2^{64} = 2^{127}$ memory

accesses (which are equivalent to $\frac{2^{127}}{9} \approx 2^{123.84}$ 9-round SEED encryptions by our estimate given in Section 4.4.3). There are a total of about $2^{63} \times 2^{64} = 2^{127}$ possible combinations of (u, K_9, w) and thus on average there is about only one value of K_9 in every single entry (u, w) of the table.

In fact, we can achieve greater efficiency. Recall that x is the 64-bit state immediately after the XOR operation with the round subkey K_9 in the **F** function. We generate the table as follows:

- For every possible 64-bit value x such that $x < [x \oplus (\alpha, 0)]$:
  - Compute the output difference of the **F** function corresponding to the pair of intermediate values $(x, x \oplus (\alpha, 0))$; and we denote it by y.
  - For every possible value of the round subkey K_9, store K_9 into the table indexed by $(\min\{x \oplus K_9, x \oplus (\alpha, 0) \oplus K_9\}, y)$.

This latter precomputation also takes the same amount of memory of 2^{130} bytes, but it has a time complexity of about $2^{63} \times 2 = 2^{64}$ computations of the **F** function (which is approximately equivalent to $2^{64} \times \frac{1}{9} \approx 2^{60.84}$ 9-round SEED encryptions) and about $2^{63} \times 2^{64} = 2^{127}$ memory accesses which is slightly faster than the former precomputation.

Consequently, we can retrieve the candidate K_9 with a single lookup to entry $(\min\{C^L_{i,j}, \widehat{C}^L_{i,l}\}, (\widehat{X}, \beta) \oplus C^R_{i,j} \oplus \widehat{C}^R_{i,l})$ of this table in Step 3(c)(iii) of the online attack procedure given in Section 4.4.3. As a result, the resulting attack has a total time complexity of approximately $2^{125} + \frac{2^{60} \times 2^{64} \times (1+1)}{9} + 2^{60} \times 2^{64} \times 2 \times \frac{1}{9} + \frac{2^{60} \times 2^{64} \times (1+1)}{9} + 2^{60} \times 2^{64} \times \frac{2}{9} + 2^{60} \times 2^{64} + \frac{2^{127}}{9} \approx 2^{126.26}$ 9-round SEED encryptions given 2^{60} pairs of plaintext structures with the same success probability of 99.9%. Meanwhile, it has a total time complexity of approximately $2^{124} + \frac{2^{59} \times 2^{64} \times (1+1)}{9} + 2^{59} \times 2^{64} \times 2 \times \frac{1}{9} + \frac{2^{59} \times 2^{64} \times (1+1)}{9} + 2^{59} \times 2^{64} \times \frac{2}{9} + 2^{59} \times 2^{64} + \frac{2^{127}}{9} \approx 2^{125.51}$ 9-round SEED encryptions given 2^{59} pairs of plaintext structures with the same success probability of 97.8%. The

first version is slightly faster than the corresponding version described in Section 4.4.3 and the second version is slightly slower than the corresponding version described in Section 4.4.3, but both require a dramatically larger memory.

4.4.4 (c) Note 3

Yanami and Shimoyama's 7-round attack and Sung's 8-round attack used 2^{64} counters on the attacked last round subkey to filter out the candidate with the highest number as the correct subkey. Thus, we can follow Theorem 3 of Selçuk (2008) to compute their approximate success probabilities which are roughly 68.8% and 98.1%, respectively. Note that they can achieve a higher success probability simply by considering the counters with the highest few numbers.

4.5 Summary

SEED is a 128-bit block cipher which is an ISO standard with a 128-bit secret key and a total of 16 rounds. In this chapter, we have described some 7-round differentials with probabilities slightly larger than the previously known ones on SEED and have presented a differential attack on 9-round SEED. The presented attack is theoretical and it does not threaten the security of the full SEED cipher. Nevertheless, from a cryptanalytic view, it suggests that the safety margin of SEED decreases below half of the number of rounds.

CHAPTER 5

IMPOSSIBLE DIFFERENTIAL ATTACK ON THE FULL CHAIN BLOCK CIPHER

The CHAIN block cipher has a variable block length, a variable secret key length and a variable number of rounds. It is a byte-oriented block cipher designed by Peyravian and Coppersmith. In this chapter, we falsify the feasibility of the differential attacks presented by the designers on 8-round 128-bit CHAIN block cipher. More importantly, we describe r'-round impossible differential characteristic on CHAIN for a variable block length where r denotes the minimum number of rounds to thwart both differential and linear attacks and $r' \in \{r-1, r\}$. Building on such r'-round impossible differential characteristics, we present an impossible differential attack on $(r'+2)$-round CHAIN cipher. This shows that the full CHAIN cipher is not a secure cipher that is suitable for any security applications. To the best of our knowledge, this is the first known cryptanalytic attack against CHAIN.

5.1 Introduction

The CHAIN block cipher was designed by Peyravian and Coppersmith (1999) from IBM. CHAIN is a unique iterated block cipher as it was designed based on two different structures, i.e., Feistel network and substitution-permutation network. It has a variable block length, a variable secret key length and a variable number of rounds. By analysing the avalanche effect of the cipher, Peyravian and Coppersmith suggested a possible minimum number of rounds for the cipher, denoted by r (which depends on the block length, see Section 5.2 for the exact value of r) to resist against both differential and linear attacks. We consider the version of CHAIN that uses $r+2$ rounds in this chapter. Such versions are cryptographically stronger than the full CHAIN cipher which uses r rounds.

The impossible differential attack was first used by Knudsen to attack the DEAL cipher (Knudsen, 1998) and it was later formalised by Biham et al. (1999a). The main principle underlying an impossible differential attack is to construct differentials that hold with probability 0 (namely impossible differentials) to progressively filter out the wrong keys till the right key remains. Note that an impossible differential attack is opposed to a conventional differential attack (Biham & Shamir, 1991a) as differentials with high probability are exploited to find the right key in differential cryptanalysis. The reader may refer to Section 2.1.3 (b) for a discussion on the impossible differential attack.

In this chapter, we initiate the security analysis of CHAIN. We first falsify the differential attacks presented by the designers on CHAIN with a block length of 128 bits using the conventional differential cryptanalysis proposed by Biham and Shamir (1991a). Furthermore, we prove that an attacker can always construct an r'-round impossible differentials on CHAIN for a variable block length where $r' \in \{r-1, r\}$. Building on such r'-round impossible differentials, we present an impossible differential attack on $(r'+2)$-round CHAIN cipher using two concrete versions of CHAIN, i.e., a 7-round CHAIN with a block length of 64 bits and an 8-round CHAIN with a block length of 128 bits. Table 5.1 summarises our impossible differential attacks on CHAIN where CP refers to the number of chosen plaintexts and Enc. refers to the required number of encryption operations of the relevant version of CHAIN.

Table 5.1: Main Cryptanalytic Results of Impossible Differential Cryptanalysis on CHAIN

Rounds	Data	Memory	Time	Source
8	$2^{111.42}$ CP	2^{88}Bytes	$2^{111.42}$Enc.	Section 5.5.1
7	$2^{55.42}$ CP	$2^{51.81}$Bytes	$2^{55.48}$Enc.	Section 5.5.2

Organisation. The remainder of the chapter is organised as follows. In the next section, we describe the CHAIN cipher. In Section 5.3, we falsify the previous differential analysis presented by the designers on CHAIN. In Section 5.4, we provide a generic construction of impossible differential characteristic of CHAIN for a variable block length and prove that these characteristics cover either $r-1$ or r rounds of the cipher. In Section 5.5, we illustrate the impossible differential attack on $(r+2)$-round CHAIN using two concrete examples, i.e., 64-bit CHAIN and 128-bit CHAIN. Finally, Section 5.6 concludes the chapter.

5.2 The CHAIN Block Cipher

CHAIN employs an unique structure that combines both the Feistel network and substitution-permutation network with a variable block length, a variable secret key length and a variable number of rounds. We denote the block length in terms of bits and bytes as $|B|$ and $|B|_8$ respectively. Peyravian and Coppersmith limits $|B|_8 \geq 8$ and $|B|_8/2 \bmod 2 = 0$. The main building block of CHAIN is the round function **F** which is depicted in Figure 5.1. Due to the modulo 8 reduction operation involved in the CHAIN cipher, we use slightly different notations in describing the CHAIN cipher as compared to other ciphers in this thesis. In particular, in this chapter, each number starts from 0 instead of 1. Such notations will simplify the description of the attacks presented in this chapter.

The F function. The **F** function consists of four sub-functions as follows:

- **Sub**: $\{0,1\}^{|B|/2} \rightarrow \{0,1\}^{|B|/2}$ is a substitution layer which uses four different 8×8-bit bijective S-boxes denoted as $\mathrm{S}_0, \mathrm{S}_1, \mathrm{S}_2$ and S_3 . If $X = (X_0||X_1||\cdots||X_{|B|_8/2-1})$ is a $|B|/2$-bit block of $|B|_8/2$ 8-bit words $X_0, X_1, \cdots, X_{|B|_8/2-1}$ and $Y = (Y_0||Y_1||\cdots||Y_{|B|_8/2-1})$ is a $|B|/2$-bit block of $|B|_8/2$ 8-bit words $Y_0, Y_1, \cdots, Y_{|B|_8/2-1}$, then $\mathbf{Sub}(X)$ is defined as
  - For $i = 0, 1, 2, \cdots, |B|_8/2-1$, compute $Y_i = \mathrm{S}_{i \bmod 4}(X_i)$.
  - Output $\mathbf{Sub}(X) = Y$.

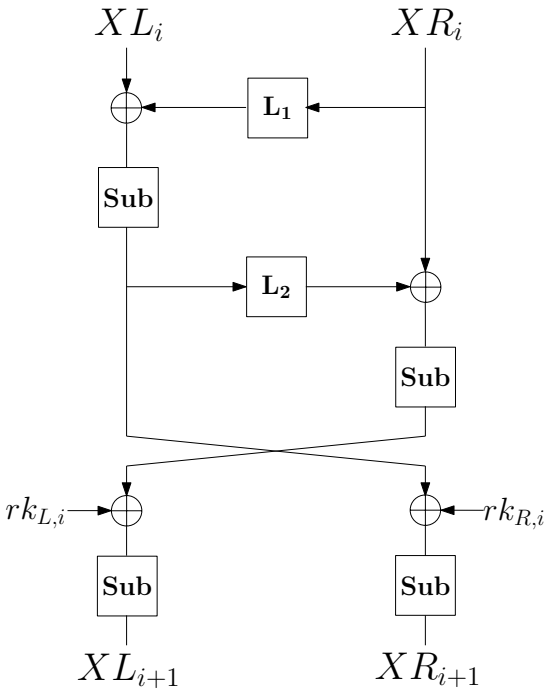


Figure 5.1: The Round Function F of CHAIN

- **Mix**: There are two linear functions $\mathbf{L_1}$ and $\mathbf{L_2}$. If X is a $|B|/2$-bit block, then $\mathbf{L_1}$ is defined as $\mathbf{L_1}(X) = (X \ggg 16) \oplus (X \ggg 8)$ and $\mathbf{L_2}$ is defined as $\mathbf{L_2}(X) = X \oplus (X \lll 24)$ where $\ggg$ and $\lll$ refer to right and left rotations of a bit string respectively.
- **Swap**: If X and Y are $|B|/2$-bit blocks, then **Swap** is defined as $\mathbf{Swap}(X,Y) = (Y,X)$.
- **ARK**: If X and Y are $|B|/2$-bit blocks, then **ARK** is defined as $\mathbf{ARK}(X,Y) = X \oplus Y$.

$\mathbf{F}: \{0,1\}^{|B|} \times \{0,1\}^{|B|} \rightarrow \{0,1\}^{|B|}$ is a non-linear function. If $X_{i-1} = (XL_{i-1}, XR_{i-1})$ is a $|B|$-bit block of two $|B|/2$-bit words XL_{i-1} and XR_{i-1}, $Y_{i-1} = (YL_{i-1}, YR_{i-1})$ is a $|B|$-bit block of two $|B|/2$-bit words YL_{i-1} and YR_{i-1}, then $\mathbf{F}(X_{i-1}, Y_{i-1})$ is defined as follows:

1) Compute $XL_i = \mathbf{Sub}(\mathbf{Sub}(XR_{i-1} \oplus \mathbf{L_2}(\mathbf{Sub}(XL_{i-1} \oplus \mathbf{L_1}(XR_{i-1})))) \oplus YL_{i-1})$.
2) Compute $XR_i = \mathbf{Sub}(\mathbf{Sub}(XL_{i-1} \oplus \mathbf{L_1}(XR_{i-1})) \oplus YR_{i-1})$.
3) Output $\mathbf{F}(X_{i-1}, Y_{i-1}) = XL_i || XR_i$.

CHAIN uses a total of r $|B|$-bit subkeys $rk_i = K_i$ for the $\mathbf{F}$ functions $i = 0, 1, 2, \cdots, r-1$. All subkeys are derived from a k-bit secret key K where k denotes the block length of K in terms of bits and each round subkey K_i is made up of two $|B|/2$-bit subkeys $K_i = (K_{i,1}, K_{i,2})$. We omit the details of the key schedule algorithm as we only focus on the encryption algorithm in this chapter.

CHAIN takes a $|B|$-bit plaintext P as input. A $|B|/2$-bit plaintext P is divided into two $|B|/2$-bit blocks $P = (P_L, P_R)$. P_L (P_R respectively) is made up of $|B|_8/2$ bytes $P_L = (P_{L,0}, P_{L,1}, P_{L,2}, \ldots, P_{L,|B|_8/2-1})$ ($P_R = (P_{R,0}, P_{R,1}, P_{R,2}, \ldots, P_{R,|B|_8/2-1})$ respectively). Its encryption algorithm $\mathtt{E}$ works as follows:

- Set $r \geq \lceil \frac{|B|_8+14}{5} \rceil$.
- Set $L_0 = P$.
- For $i = 0, 1, 2, \cdots, r-1$, compute $L_{i+1} = \mathbf{F}(L_i, K_i)$.
- Output ciphertext $C = L_r$.

For example, $r \geq 5$ for the 64-bit CHAIN block cipher (denoted as CHAIN-64) and $r \geq 6$ for the 128-bit CHAIN block cipher (denoted as CHAIN-128). In this chapter, we consider $(r+2)$-round CHAIN-64 and CHAIN-128.

5.3 On the Peyravian and Coppersmith's Differential Attacks

To analyse the strength of CHAIN-128 against differential attack, Peyravian and Coppersmith (1999) presented six different so called most promising differential attacks against CHAIN-128. For CHAIN-128, $|B| = 128$ and $|B|_8 = 16$.

Peyravian and Coppersmith presented a 6-round differential characteristic of CHAIN based on a 2-round iterated differential characteristic (i.e., the input difference is the same as the output difference). We first define the 8-bit non-zero constants a and $*$ where $*$ is not a fix 8-bit value.

Such 2-round iterated differential characteristics $\alpha \xrightarrow{2} \beta$ shown in Figure 5.2 hold with probability of 2^{-64} where $\alpha = L_0 \oplus L_0' = (aaaaaaaa, 0^8a0^8a0^8a0^8a)$ and $\beta = L_2 \oplus L_2' = (aaaaaaaa, 0^8a0^8a0^8a0^8a)$. By treating the S-boxes as completely random, $*$ will propagate to become a with the probability of $\frac{1}{256}$ after going through a S-box. We have $Pr_{\mathbf{Sub}}(******** \rightarrow aaaaaaaa) = (\frac{1}{256})^8$. When $******** \rightarrow aaaaaaaa$ holds, then $0^8*0^8*0^8*0^8* \rightarrow 0^8a0^8a0^8a0^8a$ holds with certainty.

By concatenating three 2-round iterated differential characteristics, a 6-round differential characteristic of CHAIN is formed with the probability of $(\frac{1}{256})^{24} = 2^{-192}$. To find the correct key, at least one right plaintext pair is needed. When the attacker obtains 2^{97} chosen plaintexts, a total of $\frac{2^{97} \times 2^{97}}{2} = 2^{193}$ chosen plaintext-ciphertext pairs can be generated. On average two right plaintext pairs will be obtained since the differential characteristic has a probability of 2^{-192}. Finally, the attacker can break 8-round CHAIN-128 based on a 6-round differential characteristic by exploiting the attack technique used by Biham and Shamir (1991b).

However, the S-boxes are known to be non-linear and not completely random. More precisely, for any particular input difference, not all the output differences are possible. Furthermore, the possible output differences do not occur uniformly. We run a computer simulation to compute the XOR difference distribution table (DDT) of four different S-boxes. The best differential approximation for any S-box out of the four S-boxes has the probability of $\frac{10}{256} \approx 2^{-4.678}$. By repeating the 2-round iterated differential characteristic three times, the 6-round differential characteristic will propagate through 96 active S-boxes (instead of 24 active S-boxes claimed by Peyravian and Coppersmith) with the probability of at most $2^{-4.678 \times 96} \approx 2^{-449.1}$. Note that an active S-box is defined to be an S-box that has a non-zero input difference. Similarly, an inactive S-box is defined to be an S-box that has a zero input difference. Obviously, even the attacker can gather all 2^{128} possible plaintexts, only $\frac{2^{128} \times 2^{128}}{2} = 2^{255}$ chosen plaintext-ciphertext pairs can be generated.

Since none of the 6-round differential characteristics presented by Peyravian

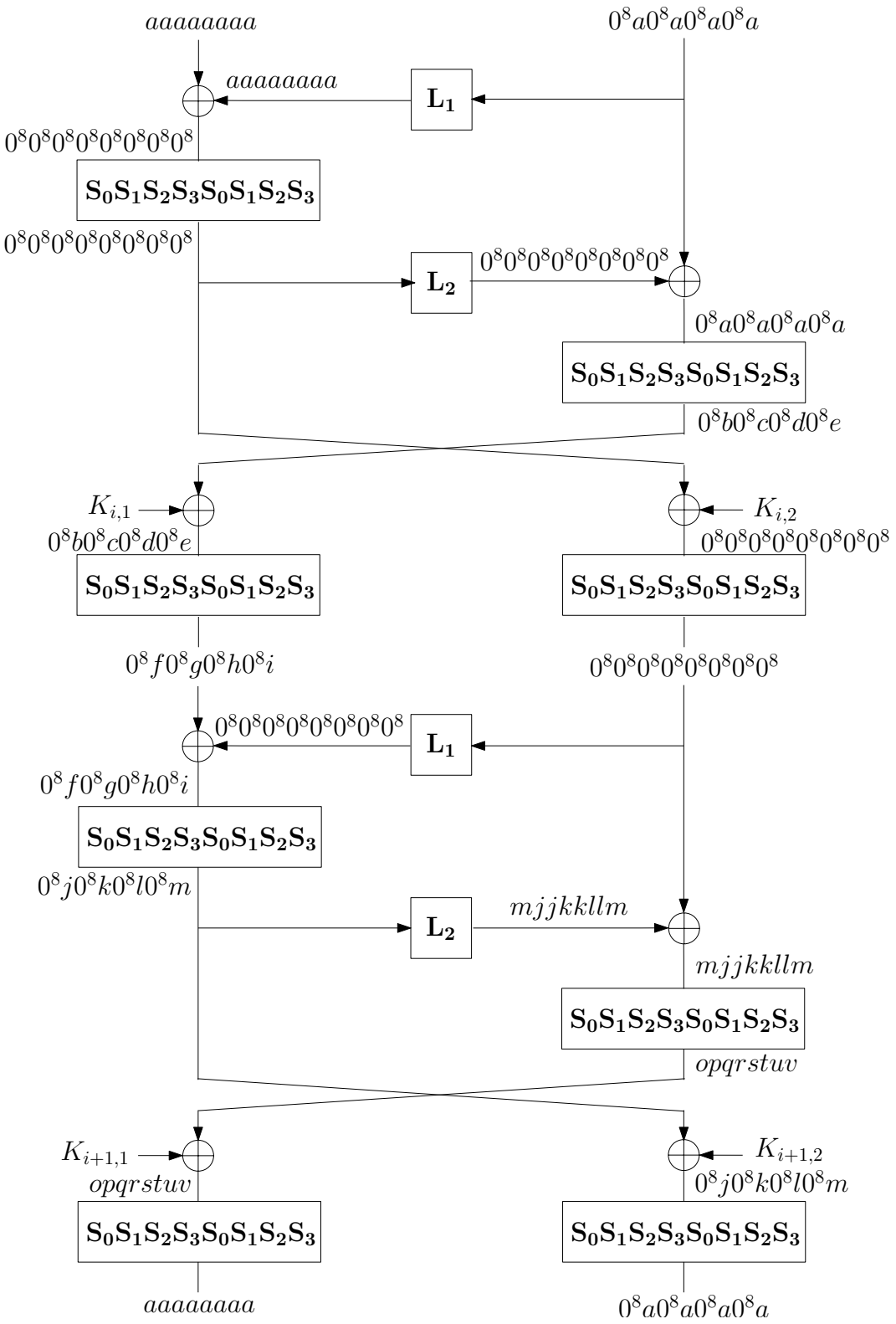


Figure 5.2: The 2-Round Iterated Differential Characteristic of CHAIN

and Coppersmith (1999) exists with the probability higher than 2^{-128}, thus no differential attacks are applicable to break 8-round CHAIN-128.

5.4 A Generic Impossible Differential Attack on $r+2$-Round CHAIN

In this section, we show that CHAIN is vulnerable to the impossible differential attack even if the number of rounds is increased from r to $r+2$. In particular, we present a way to construct a generic r'-round impossible differential characteristic of CHAIN where $r' \in \{r-1, r\}$ using the miss-in-the-middle approach introduced by Biham et al. (1999b) (see Section 2.1.3 (b) for details). Building on such r'-round impossible differential characteristics of CHAIN, we can devise an impossible differential attack against at least $r'+2$-round CHAIN.

5.4.1 Differential Characteristics with the Probability of One

To construct an impossible differential characteristic, the attacker seeks for two possibly identical differential characteristics with probability of one and concatenates such two differential characteristics such that the intermediate difference of the two differential characteristics contradict each other. We thus seek for a generic form of the differential characteristics which occur with certainty such that there is a known difference in at least one of its bytes (or 8-bit words) in either a plaintext or ciphertext pair.

As the S-boxes are bijective components, any non-zero input difference that propagates through the S-boxes will necessarily result in a non-zero output difference. Since we will primarily be interested in an arbitrary difference (i.e., to determine if any difference is present), we ignore the effect of the S-boxes in our subsequent discussion. Observe that the XOR of the round keys is canceled out when we consider both input and output XOR differences. Consequently, it suffices to examine the $\mathbf{L_1}, \mathbf{L_2}$ and **Swap** functions to determine how the difference in each byte propagates across each round.

In view of these observations, the input difference in round $i+1$ (denoted as ΔL_{i+1}) can be expressed in terms of the input difference in round i (denoted as ΔL_i) in

the following compact form where $L_i = (LL_i, LR_i)$.

$$\begin{aligned} \Delta LL_{i+1} &= \Delta LL_i \oplus (\Delta LL_i \lll 24) \oplus (\Delta LR_i \lll 16) \oplus (\Delta LR_i \lll 8) \oplus \Delta LR_i \oplus \\ &\quad (\Delta LR_i \ggg 8) \oplus (\Delta LR_i \ggg 16), \qquad (5.1) \\ \Delta LR_{i+1} &= \Delta LL_i \oplus (\Delta LR_i \ggg 8) \oplus (\Delta LR_i \ggg 16). \qquad (5.2) \end{aligned}$$

The relationships between the input differences in rounds i and $i+1$ can be further expressed in terms of bytes. For $j = 0, 1, 2, \cdots, |B|_8/2 - 1$, it can be defined as follows:

$$\begin{aligned} \Delta LL_{i+1,j} &= \Delta LL_{i,j} \oplus \Delta LL_{i,j+3 \bmod (|B|_8/2)} \oplus \Delta LR_{i,j-2 \bmod (|B|_8/2)} \oplus \\ &\quad \Delta LR_{i,j-1 \bmod (|B|_8/2)} \oplus \Delta LR_{i,j} \oplus \Delta LR_{i,j+1 \bmod (|B|_8/2)} \oplus \\ &\quad \Delta LR_{i,j+2 \bmod (|B|_8/2)}, \qquad (5.3) \\ \Delta LR_{i+1,j} &= \Delta LL_{i_j} \oplus \Delta LR_{i,j-1 \bmod (|B|_8/2)} \oplus \Delta LR_{i,j-2 \bmod (|B|_8/2)}. \qquad (5.4) \end{aligned}$$

The following lemma investigates the bytes of ΔLL_0 and ΔLR_0 involved in ΔLR_i after i rounds of encryption.

Lemma 5.1. *After i rounds of encryption (for $i \geq 2$), $\Delta LR_{i,j}$ will involve the following bytes of ΔLL_0 and ΔLR_0 :*

- $\Delta LL_{0,k} : k = j - 2i + 2 \bmod (|B|_8/2), j - 2i + 3 \bmod (|B|_8/2), \ldots, j, j+1 \bmod (|B|_8/2), \ldots, j + 3i - 7 \bmod (|B|_8/2), j + 3i - 6 \bmod (|B|_8/2)$ *and* $k = j + 3i - 3 \bmod (|B|_8/2)$*;*
- $\Delta LR_{0,k} : k = j - 2i \bmod (|B|_8/2), j - 2i + 1 \bmod (|B|_8/2), \ldots, j, j+1 \bmod (|B|_8/2), \ldots, j + 3i - 4 \bmod (|B|_8/2)$.

Besides, $\Delta LL_{0,j-2i+2 \bmod (|B|_8/2)}$, $\Delta LL_{0,j+3i-3 \bmod (|B|_8/2)}$ and $\Delta LR_{0,j-2i \bmod (|B|_8/2)}$ will be involved in $\Delta LR_{i,j}$ exactly once.

Proof. This lemma can be proved by mathematical induction. First, Equations (5.3) and (5.4) easily yields:

$$\Delta LR_{1,j} = \Delta LL_{0,j} \oplus \Delta LR_{0,j-1 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-2 \bmod (|B|_8/2)}, \tag{5.5}$$

$$\begin{aligned}
\Delta LR_{2,j} &= \Delta LL_{1,j} \oplus \Delta LR_{1,j-1 \bmod (|B|_8/2)} \oplus \Delta LR_{1,j-2 \bmod (|B|_8/2)} \\
&= \Delta LL_{0,j} \oplus \Delta LL_{0,j+3 \bmod (|B|_8/2)} \oplus \bigoplus_{k=j-2 \bmod (|B|_8/2)}^{j+2 \bmod (|B|_8/2)} \Delta LR_{0,k} \\
&\quad \oplus \Delta LL_{0,j-1 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-2 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-3 \bmod (|B|_8/2)} \\
&\quad \oplus \Delta LL_{0,j-2 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-3 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-4 \bmod (|B|_8/2)} \\
&= \Delta LL_{0,j-2 \bmod (|B|_8/2)} \oplus \Delta LL_{0,j-1 \bmod (|B|_8/2)} \oplus \Delta LL_{0,j} \\
&\quad \oplus \Delta LL_{0,j+3 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-4 \bmod (|B|_8/2)} \oplus \Delta LR_{0,j-3 \bmod (|B|_8/2)} \\
&\quad \oplus \Delta LR_{0,j-2 \bmod (|B|_8/2)} \oplus \bigoplus_{k=j-3 \bmod (|B|_8/2)}^{j+2 \bmod (|B|_8/2)} \Delta LR_{0,k}.
\end{aligned} \tag{5.6}$$

Assume that the result holds for i rounds of the encryption, $i \geq 2$. Thus, $\Delta LR_{i+1,j}$ includes $\Delta LL_{1,k_1}$ for $k_1 = j-2i+2 \bmod (|B|_8/2)$ to $j+3i-6 \bmod (|B|_8/2)$ together with $k_1 = j+3i-3 \bmod (|B|_8/2)$ as well as $\Delta LR_{1,k_2}$ for $k_2 = j-2i \bmod (|B|_8/2)$ to $j+3i-4 \bmod (|B|_8/2)$. Substituting $\Delta LL_{1,k}$ and $\Delta LR_{1,k}$ in terms of ΔLL_0 and ΔLR_0, the following bytes will be involved:

- $\Delta LL_{0,k}, k = k_1, k_1+3 \bmod (|B|_8/2)$ and $k = k_2$,
- $\Delta LR_{0,k}, k = k_1-2 \bmod (|B|/2), k_1-1 \bmod (|B|_8/2), k_1, k_1+1 \bmod (|B|_8/2), k_1+2 \bmod (|B|_8/2), k_2-1 \bmod (|B|_8/2), k_2-2 \bmod (|B|_8/2)$.

Hence, $\Delta LL_{0,j-2i \bmod (|B|_8/2)}$ comes from $\Delta LR_{1,j-2i \bmod (|B|_8/2)}$. Meanwhile, $\Delta LL_{0,j+3i \bmod (|B|_8/2)}$ comes from $\Delta LL_{1,j+3i-3 \bmod (|B|_8/2)}$. Our result now follows by our induction hypothesis. □

Lemma 5.2. *Let* $\Delta L_0 = (\Delta LL_0, \Delta LR_0) = (d, 0, \ldots, 0)$ *be a plaintext pair with a nonzero difference* d *in byte 0. Let* $n_1 = \lfloor \frac{|B|_8+14}{10} \rfloor$ *and* $n_2 = \lfloor \frac{|B|_8+12}{10} \rfloor$. *Then* $\Delta LR_{n_1, 2n_1-2} = d'$

and $\Delta LR_{n_2,2n_2-1} = 0$ *for some nonzero byte difference* d'.

Proof. Observe that since $\Delta LL_j = 0$ for $j \neq 0$ and $\Delta LR_j = 0$ for all $0 \leq j \leq |B|_8/2-1$, for any i and $j, 0 \leq j \leq |B|_8/2-1$, $\Delta LR_{i,j} = 0$ if and only if it does not involve $\Delta LL_{0,0}$ while $\Delta LR_{i,j}$ has a known nonzero difference d' if and only if it involves $\Delta LL_{0,0}$ exactly once. By Lemma 5.1, $\Delta LR_{j,n_2}$ includes the bytes $\Delta LL_{0,k}, k = j - 2n_2 + 2 \bmod (|B|_8/2), \ldots, j, \ldots, j + 3n_2 - 6 \bmod (|B|_8/2)$, and $k = j + 3n_2 - 3 \bmod (|B|_8/2)$. Let $j = 2n_2 - 1$, where $n_2 = \lfloor \frac{|B|_8+12}{10} \rfloor$. Then $k = 1, \ldots, 5n_2 - 7$ and $k = 5n_2 - 4 \bmod (|B|_8/2)$. Since $5n_2 - 7 = 5\lfloor \frac{|B|_8+12}{10} \rfloor - 7 \leq |B|_8/2 - 1 < |B|_8/2$ and $5n_2 - 4 = 5\lfloor \frac{|B|_8+12}{10} \rfloor - 4 > 5(\frac{|B|_8+12}{10}) - 1 - 4 = |B|_8/2 + 1$, $\Delta LR_{n_2,j}$ does not involve $\Delta LL_{0,0}$ at all. Consequently, $\Delta LR_{n_2,2n_2-1} = 0$. Similarly, $\Delta LR_{n_1,2n_1-2}$ involves the bytes $\Delta LL_{0,k}, k = 0, 1, \ldots, 5n_1 - 8$ and $k = 5n_1 - 5 \bmod (|B|_8/2)$. It is easy to check that $5n_1 - 8 < |B|_8/2$ and $5n_1 - 5 > |B|_8/2$ so that $\Delta LR_{n_1,2n_1-2}$ involves $\Delta LL_{0,0}$ exactly once. This concludes our proof. □

We obtain similar results in the reverse direction. The following lemma can be proved using the same approach as above and thus it is omitted.

Lemma 5.3. *Let* n_1 *and* n_2 *be as defined in Lemma 5.2. Further, let* d *denote an arbitrary nonzero byte difference.*

1. *If* $\Delta L_{n_2} = (d, 0, \ldots, 0)$, *then after* n_2 *rounds of decryption,* $\Delta LR_{0,2n_2-1} = 0$.
2. *If* $\Delta L_{n_1} = (d, 0, \ldots, 0)$, *then after* n_1 *rounds of decryption* $\Delta LR_{0,0} = d'$ *for some nonzero difference* d'.

With the aid of Lemmas 5.2 and 5.3, an impossible differential characteristic can be constructed in the following theorem.

Theorem 5.1. *For the CHAIN cipher with block size* $|B|_8$, *there exists an* r'*-round impossible differential characteristic, where* $r' = \lfloor \frac{|B|_8+12}{10} \rfloor + \lfloor \frac{|B|_8+24}{10} \rfloor$.

Proof. We construct a difference characteristic S having n_1 rounds in the forward direction and a difference characteristic T having n_2 rounds in the backward direction, where both S and T occur with probability 1 (refer to the preceding section) as follows:

1. First, we construct S. Let $\Delta L_0 = (\Delta LL^0, \Delta LR^0)$, where $\Delta LL_0 = (0, d, d, 0, \ldots, 0)$ and $\Delta LR_0 = (d, 0, \ldots, 0)$, d denoting some nonzero byte difference. It is easy to verify that $\Delta L_1 = (\Delta LL_1, \Delta LR_1)$, where $\Delta LL_1 = (d_1, 0, \ldots, 0)$ and $\Delta LR_1 = (0, 0, \ldots, 0)$. From Lemma 5.2, after another n_1 rounds of encryption, we obtain ΔL_{n_1+1}, where $\Delta LR_{n_1+1,2n_1-2} = d'$ for some nonzero byte difference d'.

2. Consider $\Delta L_{n_2} = (d, 0, \ldots, 0)$, where d is some nonzero byte difference. By Lemma 5.3, after n_2 rounds of decryption, we obtain ΔL_0 such that $\Delta LR_{0,2n_2-1} = 0$. Let us denote this difference characteristic by **D**. Now by the structure of CHAIN, any l-byte rotation of ΔLL_{n_2} and ΔLR_{n_2} simultaneously results in the corresponding l-byte rotation for each of the ΔLL_i and $\Delta LR_i, 1 \leq i \leq n_2$. Let $l = 2n_1 - 2n_2 - 1 \bmod |B|_8/2$. By rotating each of the left and right halves of **D** by l bytes, we obtain a difference characteristic T beginning with $\Delta L_{n_2} = (\Delta LL_{n_2}, \Delta LR_{n_2})$ and terminating in $\Delta L_0 = (\Delta LL_0, \Delta LR_0)$ such that $\Delta LL_{n_2,l} = d, \Delta LL_{n_2,j} = 0, j \neq l, \Delta LR_{n_2,j} = 0$ and $\Delta LR_{0,2n_1-2} = 0$.

Consequently, S and T satisfy the required properties, yielding an impossible differential characteristic having $n_1 + n_2 + 1$ rounds. □

We remark that different characteristics may be obtained by varying l in the above theorem, as long as for some j, byte j of ΔL_{n_1+1} and ΔL_0 contradict. Further, it is straightforward to verify that r' is either equal to $r-1$ or r. For example, both r' and r are equal when $|B|_8 = 8$ or 16.

By adding a round before S and at the end of T each, we can exploit the given impossible differential characteristic to launch an impossible differential attack for

$r'+2 = n_1+n_2+3$ rounds of CHAIN, thereby obtaining the correct key bytes for a certain portion of rk_0 and $rk_{r'+1}$.

5.5 Impossible Differential Attacks on CHAIN-64 and CHAIN-128

In this section, we demonstrate the application of the generic attack given in Theorem 5.1 on 7-round CHAIN-64 and 8-round CHAIN-128. In each of these concrete examples, we break $r' = n_1+n_2+3 = r+2$ rounds of CHAIN using the impossible differential characteristic presented in Section 5.4.

5.5.1 Impossible Differential Attack on 8-Round CHAIN-128

CHAIN-128 has a 128-bit block length and a recommended number of $r \geq \lceil \frac{|B|_8+14}{5} \rceil \geq 6$ rounds. We consider the 128-bit version of CHAIN that uses 8 rounds in this chapter. Besides, CHAIN-128 uses eight 128-bit subkeys $rk_i = K_i$ for the **F** functions ($i = 0,1,\cdots,7$). Each subkey K_i is made up of two 64-bit subkeys $K_i = (K_{i,1}, K_{i,2})$. Lastly, each 64-bit subkey $K_{i,j}$ is made up of eight bytes $K_{i,j} = (K_{i,j,0}, K_{i,j,1}, K_{i,j,2}, K_{i,j,3}, K_{i,j,4}, K_{i,j,5}, K_{i,j,6}, K_{i,j,7})$ for $j = 1,2$.

Two differential characteristics with probability of 1, S and T, constructed using Theorem 5.1 are defined as follows.

$$\begin{aligned} S &= 0^8 aa 0^8 0^8 0^8 0^8 0^8 a 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 \xrightarrow{4} ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 ?^8 y ?^8 ?^8 ?^8, \\ T &= \Delta_1 x ?^8 ?^8 z 0^8 xxyz 0^8 0^8 x 0^8 0^8 \xrightarrow{2} s 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8 \end{aligned}$$

where the question marker $?^8$ represents a binary string of 8 question marker ? (? represents an indetermine bit; alternatively, $?^8$ represents an indetermine byte), a, s, x, y, z and Δ_1 represent known non-zero 8-bit words (or a known non-zero byte). The differential characteristics S and T are illustrated in Figures 5.3 and 5.4.

Since the output difference of the 4-round differential characteristic S contradicts with the input difference of the 2-round differential characteristic T (see byte $LR_{0,4}$), thus we have a 6-round impossible differential characteristic $\mathbf{D_3}$ $0^8 aa 0^8 0^8 0^8 0^8 0^8$

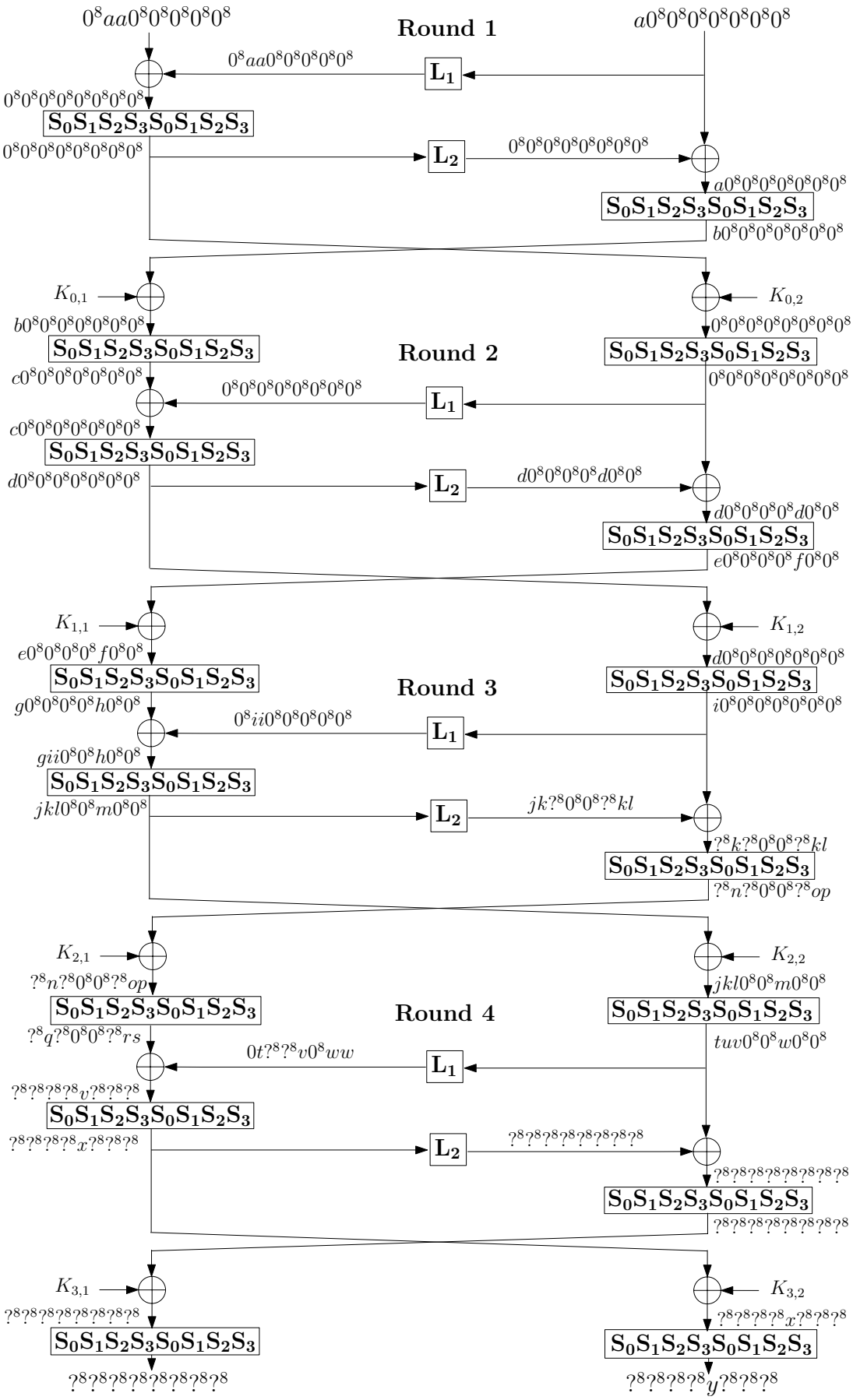


Figure 5.3: The 4-Round Differential Characteristic S on CHAIN-128 with Probability of One

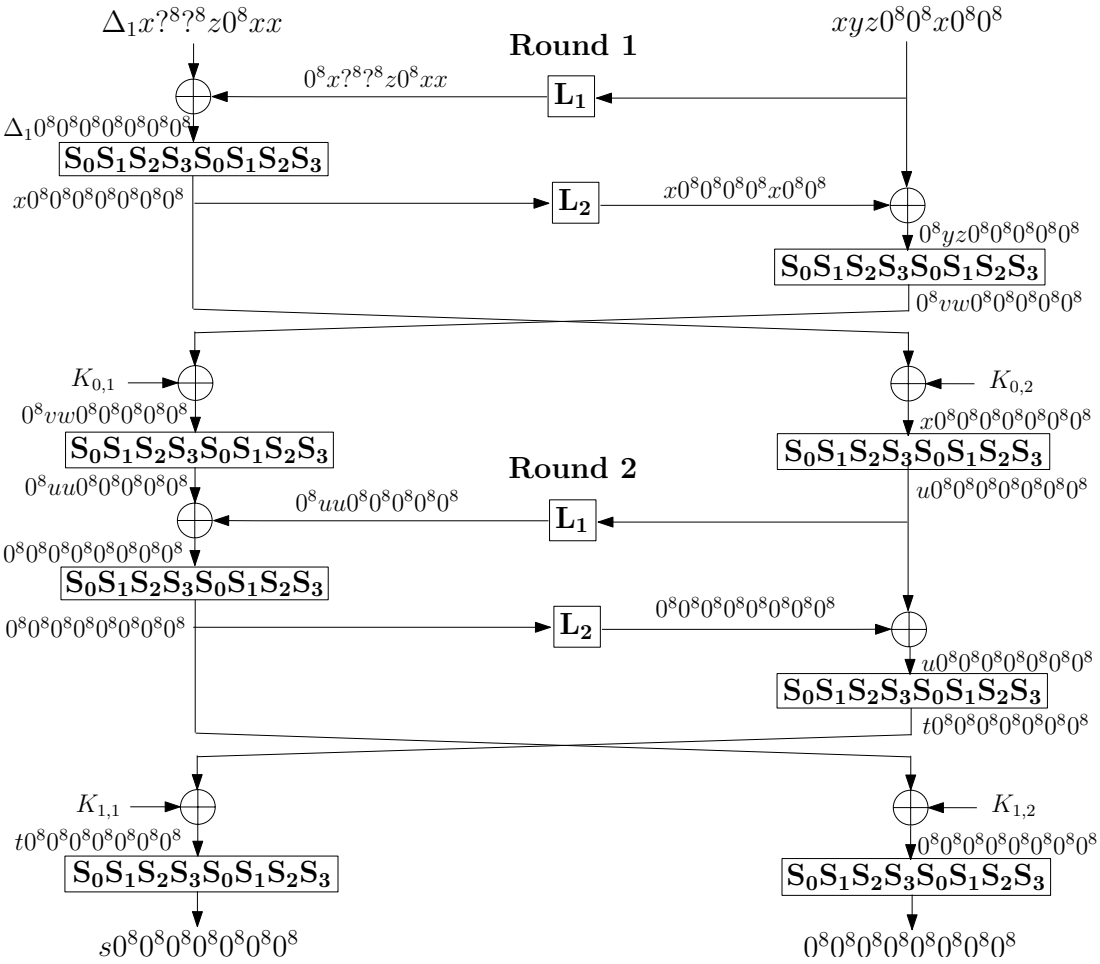


Figure 5.4: The 2-Round Differential Characteristic T on CHAIN-128 with Probability of One

$a0^80^80^80^80^80^80^80^8 \overset{6}{\not\to} s0^80^80^80^80^80^80^80^80^80^80^80^80^80^80^80^8$. Note that both S and T occur with a probability of 1 and $\mathbf{D_3}$ will hold with a probability of 0. By adding one round before and after $\mathbf{D_3}$, we can thus attack 8-round CHAIN-128.

To form such an impossible differential distinguisher, observe that the plaintext XOR difference ΔP must have the form of $\mathbf{S_1^{-1}}(\alpha)\alpha(\alpha\oplus\beta)(\beta\oplus\gamma)\gamma0^8\alpha\alpha\alpha\beta\gamma0^80^8\alpha0^8$ 0^8 and the ciphertext XOR difference ΔC must have the form of $\Delta_30^80^80^80^8\Delta_40^80^8\Delta_50^8$ $0^80^80^80^80^80^8$. Such forms are obtained by decrypting the input difference of $\mathbf{D_3}$ and encrypting the output difference of $\mathbf{D_3}$.

5.5.1 (a) *Attack Procedure*

We can now devise the following impossible differential attack procedure to break the full 8-round CHAIN-128 illustrated in Figure 5.5.

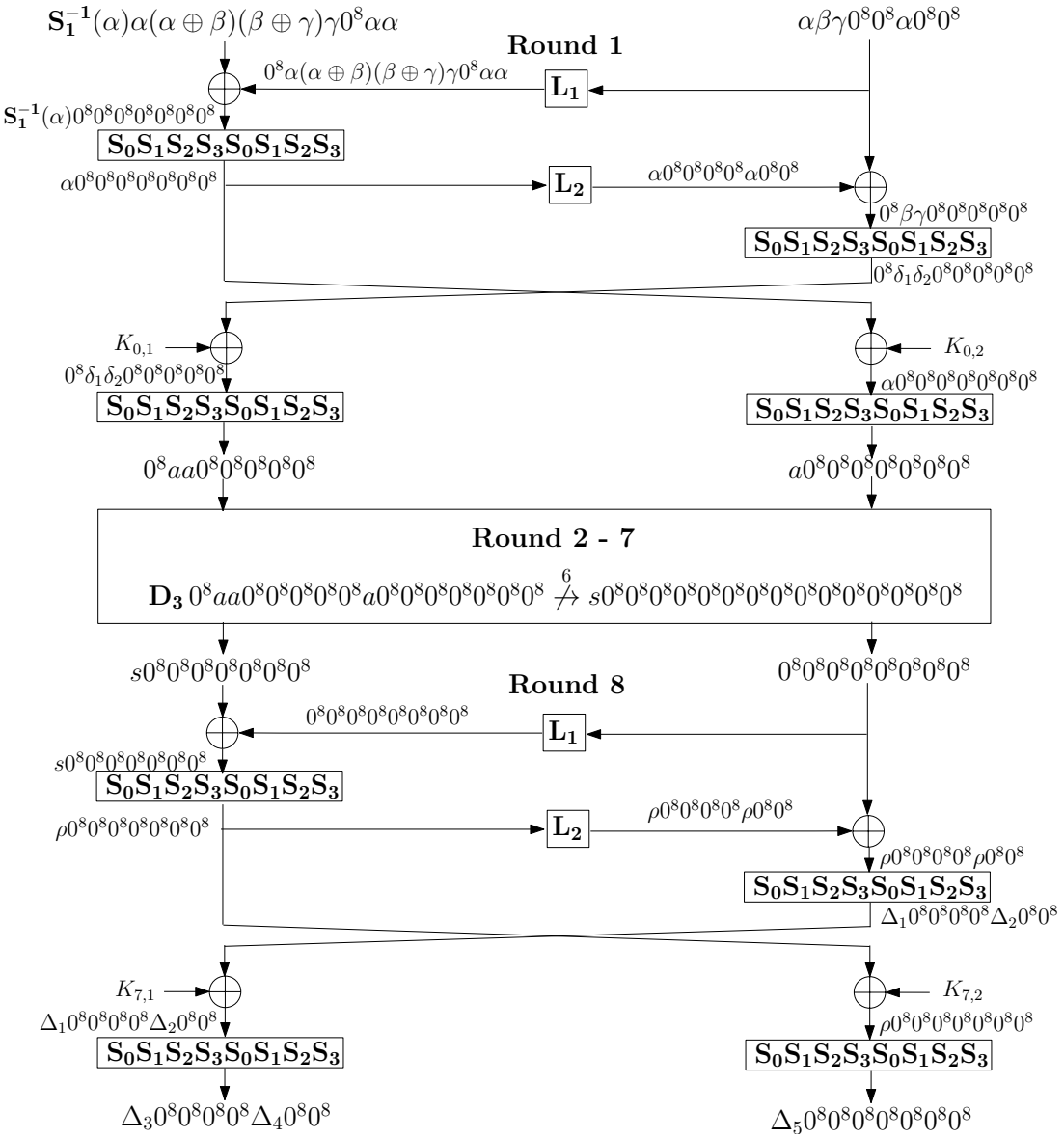


Figure 5.5: The Impossible Differential Attack on 8-Round CHAIN-128

1) Choose 2^{ϕ} structures $\mathscr{S}_i$ (a specific value of ϕ will be given below and for $i = 1,2,\cdots,2^{\phi}$) where a structure is defined to be a set of 2^{88} plaintexts $P_{i,j}$ (for $j = 1,2,\cdots,2^{88}$) with $P_{L,0}, P_{L,1}, P_{L,2}, P_{L,3}, P_{L,4}, P_{L,6}, P_{L,7}, P_{R,0}, P_{R,1}, P_{R,2}$ & $P_{R,5}$)

take all the possible 8-bit values and $P_{L,5}, P_{R,3}, P_{R,4}, P_{R,6}$ & $P_{R,7}$ are fixed to a certain value. In a chosen plaintext attack scenario, obtain all the ciphertexts for the 2^{88} plaintexts in each of the 2^{ϕ} structures. We denote $C_{i,j}$ as the ciphertext for plaintext $P_{i,j}$.

2) For each structure, perform the following sub-steps:

   a) For $1 \leq l \leq 2^{88}$, identify all plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ where $P_{i,j} \oplus \widehat{P}_{i,l}$ is equal to $\mathbf{S_1^{-1}}(\alpha)\alpha(\alpha \oplus \beta)(\beta \oplus \gamma)\gamma 0^8 \alpha\alpha\alpha\beta\gamma 0^8 0^8 \alpha 0^8 0^8$ and $C_{i,j} \oplus \widehat{C}_{i,l}$ is equal to $\Delta_3 0^8 0^8 0^8 0^8 \Delta_4 0^8 0^8 \Delta_5 0^8 0^8 0^8 0^8 0^8 0^8 0^8 0^8$. Note that $\alpha, \beta, \gamma, \Delta_3, \Delta_4$ and Δ_5 represent a non-zero byte and they may or may not equal to each other.

   b) Guess a value for $K_{0,1,1}$ & $K_{0,2,0}$ and perform the following sub-steps:

      (i) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$. compute $\mathbf{g_1}(P, \widehat{P}) = \mathbf{f_1}(P) \oplus \mathbf{f_1}(\widehat{P}) \oplus \mathbf{f_2}(P) \oplus \mathbf{f_2}(\widehat{P})$ such that $\mathbf{f_1}(X) = \mathbf{S_0}(\mathbf{S_0}(X_{L,0} \oplus X_{R,6} \oplus X_{R,7}) \oplus K_{0,2,0})$ and $\mathbf{f_2}(X) = \mathbf{S_1}(\mathbf{S_1}(\mathbf{S_1}(X_{L,1} \oplus X_{R,0} \oplus X_{R,7}) \oplus \mathbf{S_0}(X_{L,4} \oplus X_{R,2} \oplus X_{R,3}) \oplus X_{R,1}) \oplus K_{0,1,1})$. Then, discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if $\mathbf{g_1}(P, \widehat{P}) \neq 0$.

      (ii) Guess a value for $K_{0,1,2}$ and perform the following sub-steps:

         A) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$, compute $\mathbf{f_3}(P) \oplus \mathbf{f_3}(\widehat{P})$ such that $\mathbf{f_3}(X) = \mathbf{S_2}(\mathbf{S_2}(\mathbf{S_2}(X_{L,2} \oplus X_{R,0} \oplus X_{R,1}) \oplus \mathbf{S_1}(X_{L,5} \oplus X_{R,3} \oplus X_{R,4}) \oplus X_{R,2}) \oplus K_{0,1,2})$. Discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if $\mathbf{f_3}(P) \oplus \mathbf{f_3}(\widehat{P}) \oplus \mathbf{f_1}(P) \oplus \mathbf{f_1}(\widehat{P}) \neq 0$.

         B) Guess a value for $K_{7,1,0}$ & $K_{7,2,0}$ and perform the following sub-steps (see Figure 5.6 for details):

            I) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$, compute $\mathbf{g_2}(C, \widehat{C}) = \mathbf{f_4}(C) \oplus \mathbf{f_4}(\widehat{C}) \oplus \mathbf{f_5}(C) \oplus \mathbf{f_5}(\widehat{C})$ where $\mathbf{f_4}(X) = \mathbf{S_0^{-1}}(\mathbf{S_0^{-1}}(X_{L,0}) \oplus K_{7,1,0})$ and $\mathbf{f_5}(X) = \mathbf{S_0^{-1}}(X_{R,0}) \oplus K_{7,2,0}$. Discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if $\mathbf{g_2}(C, \widehat{C}) \neq 0$.

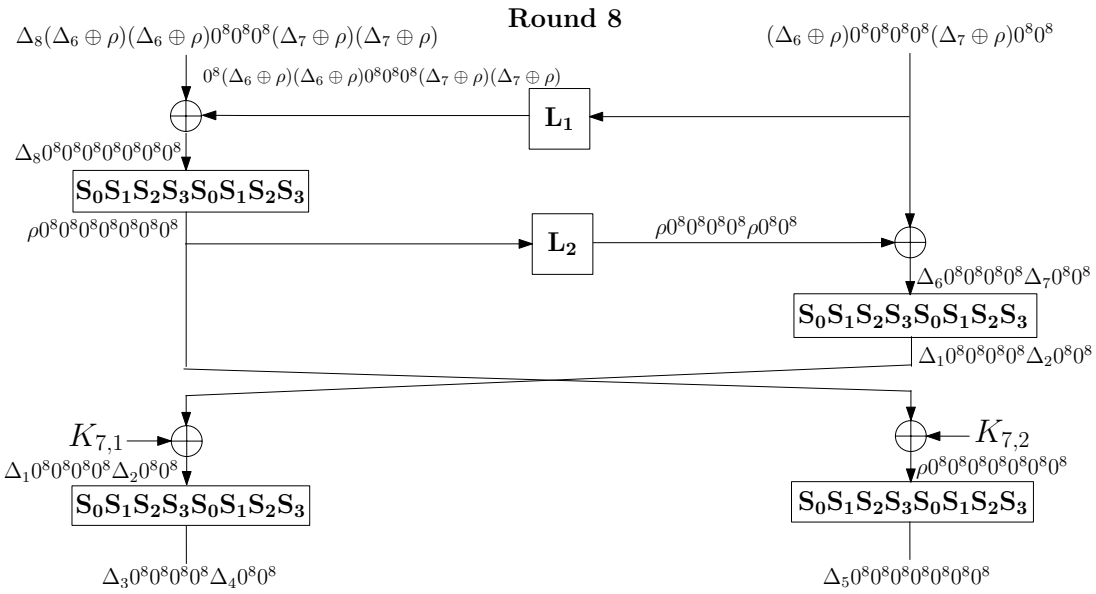


Figure 5.6: One-Round Decryption of Round 8 of CHAIN-128

II) Guess a value for $K_{7,1,5}$. For each remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$, compute $\mathbf{f_6}(C) \oplus \mathbf{f_6}(\widehat{C})$ such that $\mathbf{f_6}(X) = \mathbf{S_1^{-1}}(\mathbf{S_1^{-1}}(X_{L,5}) \oplus K_{7,1,5})$. If $\mathbf{f_6}(C) \oplus \mathbf{f_6}(\widehat{C}) = \mathbf{f_5}(C) \oplus \mathbf{f_5}(\widehat{C})$, then the guess of six subkey bytes $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{7,1,0}, K_{7,2,0}$ and $K_{7,1,5})$ is wrong and thus can be filtered out.

Given a sufficient number of chosen plaintext-ciphertext pairs, all the wrong subkey bytes will be discarded. Notice that different impossible characteristics can be utilized to filter out the other wrong subkey bytes for Round 8 by using a different differential characteristic T (see the impossible differential attack against CHAIN-64 in Section 5.5.2). Once the round subkey K_7 is successfully recovered, Round 8 of CHAIN-128 can be removed and the same idea can be used to recover the remaining subkeys for Rounds 1 to 7.

5.5.1 (b) Attack Complexity

The attack requires $2^{\phi} \times 2^{88}$ chosen plaintexts. Step 1 has a time complexity of $2^{\phi+88}$ 8-round CHAIN-64 encryptions. For a structure of 2^{88} chosen plaintexts, Step 1 has a memory complexity of $2^{88} \times (11+16)$ bytes $\approx 2^{92.76}$ bytes since five bytes of

all plaintexts are fixed to a certain value. The plaintext-ciphertext pairs are stored in a table indexed by the ciphertext.

In Step 2(a), for a structure of 2^{88} chosen plaintexts, a total of $\binom{2^{88}}{2} \approx 2^{175}$ chosen plaintext-ciphertext quartets can be generated. However, the expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{175} \times 2^{-56} \times 2^{-104} \times \frac{100}{256} \approx 2^{13.64}$. Notice that according to the difference distribution table, for the S-box $\mathbf{S_0}$, there are expected 100 out of 256 possible non-zero input differences x such that $\mathbf{S_0}(x) = y$ for a fixed non-zero output difference y. Thus, the memory complexity of Step 2 is $2^{13.64} \times (11+16+11+3)$ bytes $\approx 2^{19.00}$ bytes since 13 bytes of both ciphertexts are similar. Since there is a 161.36-bit filtering condition in Step 2 for the differences in the plaintext-ciphertext quartet, thus Step 2 has a time complexity of about $2^{13.64}$ memory accesses.

Next, we analyse Step 2(b) for a structure of $2^{13.64}$ chosen plaintext-ciphertext quartets. Since there exists 2^{16} possible values of $(K_{0,1,1}, K_{0,2,0})$, Step 2(b)(i) requires $2^{13.64} \times 2^{16} \times (23$ XOR $+12$ S-box lookup) operations. Since one-round CHAIN-128 encryption requires 3 rotations, 6 XOR and 32 S-box table lookup operations, thus we consider the sum of 23 XOR and 12 S-box lookup operations equals $\frac{35}{41}$ of the time complexity of one-round CHAIN-128 encryption. Step 2(b)(i) has a time complexity of $2^{13.64} \times 2^{16} \times \frac{35/41}{8} \approx 2^{26.42}$ 8-round CHAIN-128 encryptions. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{13.64} \times 2^{-8} = 2^{5.64}$. Step 2(b)(ii)(A) requires 23 XOR and 12 S-box table lookup operations (which should be less than $\frac{35}{41}$ of the complexity of one-round CHAIN-128 encryption) for each guess of 2^8 possible values of $K_{0,1,2}$, thus it has a time complexity of $2^8 \times 2^{5.64} \times \frac{35/41}{8} \approx 2^{10.42}$ 8-round CHAIN-128 encryptions. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{5.64} \times 2^{-8} = 2^{-3.64}$. Subsequently, Step 2(b)(ii)(B)(I) requires 7 XOR and 6 inverse of S-box table lookup operations (which should be less than $\frac{13}{41}$ of the complexity of one-round CHAIN-128 encryption) for each guess of 2^{16} possible values of $(K_{7,1,0}, K_{7,2,0})$, thus it has a time complexity of $2^{16} \times 2^{-3.64} \times \frac{13/41}{8} \approx 2^{7.71}$ 8-round CHAIN-128 encryptions. The expected remaining chosen plaintext-ciphertext

quartets in a structure is $2^{-3.64} \times 2^{-8} = 2^{-11.64}$. Finally, Step 2(b)(ii)(B)(II) requires 6 XOR and 6 inverse operations of S-box table lookup operations (which should be less than $\frac{12}{41}$ of the complexity of one-round CHAIN-128 encryption) for each guess of 2^8 possible values of $K_{7,1,5}$, thus it has a time complexity of $2^8 \times 2^{-11.64} \times \frac{12/41}{8} \approx 2^{-8.42}$ 8-round CHAIN-128 encryptions.

In total, the impossible attack has a time complexity of approximately $2^{\phi+88} + 2^{\phi} \times (2^{26.42} + 2^{10.42} + 2^{7.71} + 2^{-8.42}) \approx 2^{\phi+88}$ 8-round CHAIN-128 encryptions and $2^{\phi+13.64}$ memory accesses.

As mentioned in Section 3.3.3 (c), the question on how many memory accesses (table lookups) are equivalent to one CHAIN-128 encryption in terms of time depends closely on the used platform and CHAIN-128 implementation as well as the storage location of the sorted table. In theoretical block cipher cryptanalysis, it is usually assumed by default that a table is stored in an ideal place, RAM say, like an S-box table and it takes an almost constant time to access an entry in a sorted table independent of the number of entries. Thus, an extremely conservative estimate is that 8 memory accesses equal a 8-round CHAIN-128 encryption in terms of time assuming that the **F** function with a round subkey is precomputed in a table and is equivalent to one memory access by neglecting the computational complexity for the key schedule. Thus, one round is equivalent to one memory access. As a consequence, the total time complexity of the impossible differential attack is $2^{\phi+88} + \frac{2^{\phi+13.64}}{8} \approx 2^{\phi+88}$ 8-round CHAIN-128 encryptions.

The attack succeeds if the expected number of subkey bytes that will not be filtered out after performing the above attack procedure is less than 1 as one out of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{7,1,0}, K_{7,2,0}, K_{7,1,5})$ must be correct. In the beginning of Step 2(b)(ii)(B)(II),the expected remaining chosen plaintext-ciphertext quartets in all the structures is $2^{\phi-11.64}$. Observe that given any guess of $K_{7,1,5}$, the last condition is fulfilled with the probability of 2^{-8} and thus the probability that each of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{7,1,0}, K_{7,2,0}, K_{7,1,5})$

is filtered out using one of $2^{\phi-11.64}$ possible chosen plaintext-ciphertext quartets is 2^{-8}. It follows that the expected number of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{7,1,0}, K_{7,2,0}, K_{7,1,5})$ that will not be filtered out after performing the above attack procedure is $2^{48} \times (1-2^{-8})^{2^{\phi-11.64}}$. Thus, by letting $\phi = 23.42$, only one of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{7,1,0}, K_{7,2,0}, K_{7,1,5})$ will remain and it must be the right subkey.

As a conclusion, the impossible differential attack has a time complexity of $2^{\phi+88} = 2^{111.42}$ 8-round CHAIN-128 encryption, a memory complexity of 2^{48} bytes (which is dominated by the step in storing the list of filtered wrong key bytes) and a data complexity of $2^{\phi+88} = 2^{111.42}$ chosen plaintexts.

5.5.2 Impossible Differential Attack on 7-Round CHAIN-64

Different with CHAIN-128, CHAIN-64 has a 64-bit block length and a recommended number of $r \geq \lceil \frac{|B|_8+14}{5} \rceil \geq 5$ rounds. We consider the 64-bit version of CHAIN that uses 7 rounds in this chapter. Besides, CHAIN-64 uses seven 64-bit subkeys $rk_i = K_i$ for the $\mathbf{F}$ functions $i = 0, 1, \cdots, 6$. Each subkey K_i is made up of two 32-bit subkeys $K_i = (K_{i,1}, K_{i,2})$. Lastly, each 32-bit subkey $K_{i,j}$ is made up of four bytes $K_{i,j} = (K_{i,j,0}, K_{i,j,1}, K_{i,j,2}, K_{i,j,3})$ for $j = 1, 2$.

Two differential characteristics with probability of 1 S and T constructed are defined as follows:

$$\begin{aligned} S &= 0^8 aa0^8 a0^8 0^8 0^8 \xrightarrow{3} ?^8 ?^8 ?^8 mn?^8 o0, \\ T &= ?^8 ?^8 x?^8 0^8 x?^8 z \xrightarrow{2} 0^8 s0^8 0^8 0^8 0^8 0^8 0^8 \end{aligned}$$

where the question marker $?^8$ represents a binary string of 8 question marker ? (? represents an indeterminate bit; alternatively, $?^8$ represents an indeterminate byte), a, l, m, n, s, x and z represent known non-zero 8-bit words (or a known non-zero byte). The differential characteristics S and T are illustrated in Figures 5.7 and 5.8.

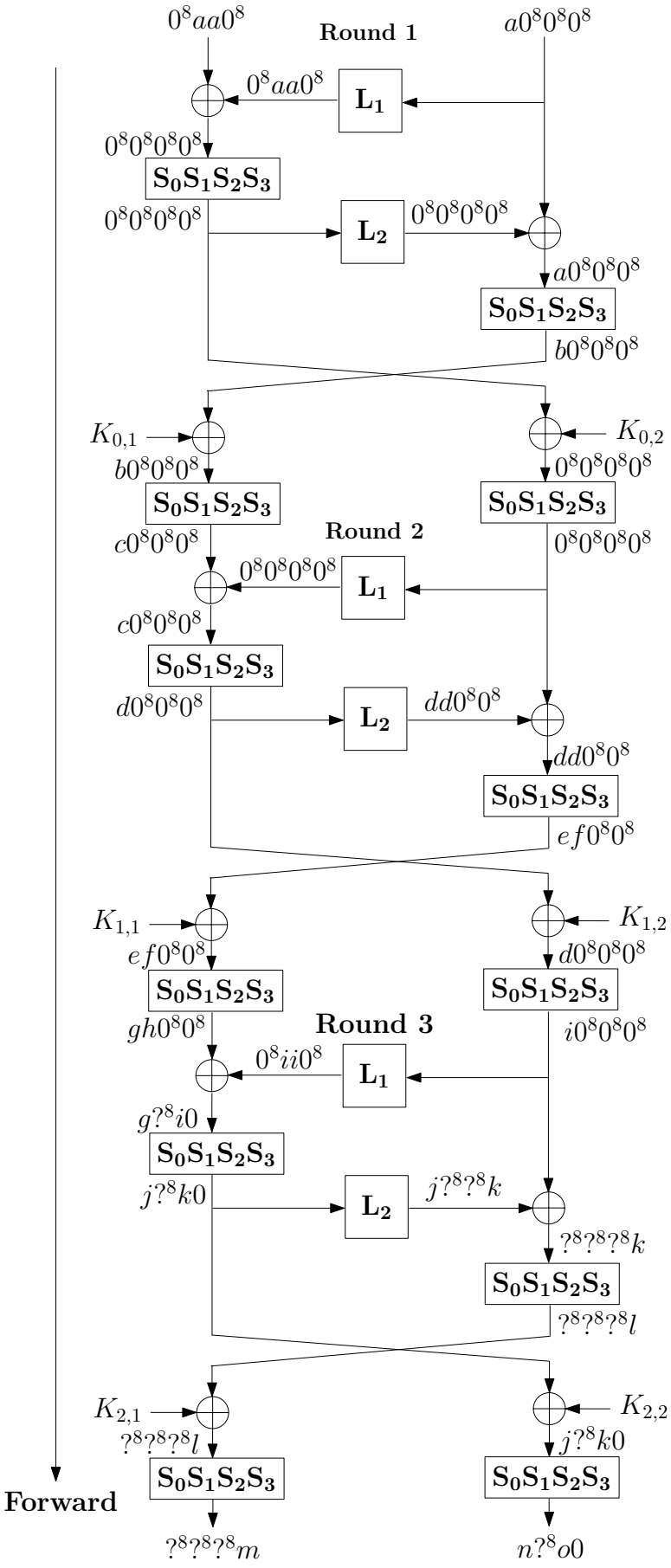


Figure 5.7: The 3-Round Differential Characteristic S with Probability of One

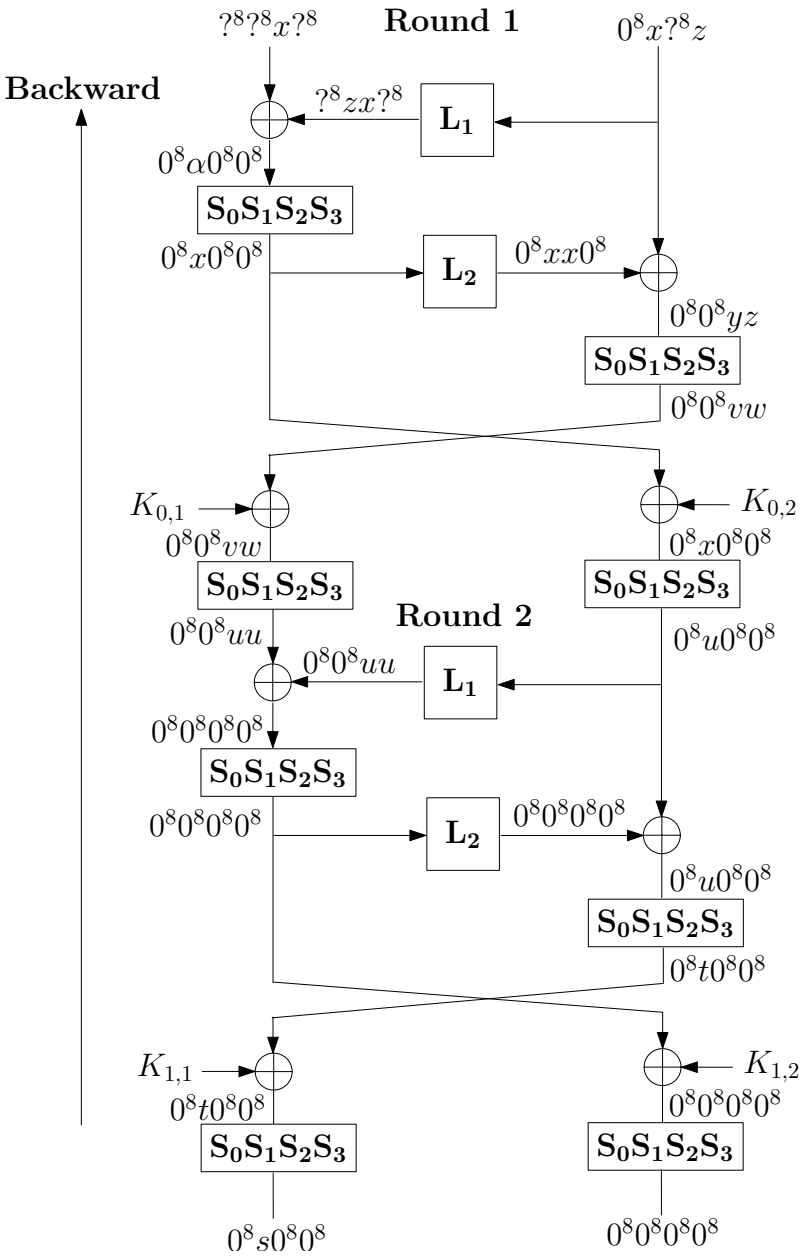


Figure 5.8: The 2-Round Differential Characteristic T with Probability of One

Notice that the differential characteristic S is constructed using Theorem 5.1 while the differential characteristic T is constructed by modifying the byte position of T given in Theorem 5.1 to show that different round subkey bytes can be recovered using slightly different differential characteristic T. One can remove the last round of CHAIN if a last round subkey can be fully recovered.

Since the output difference of the 3-round differential characteristic S contradicts with the input difference of the 2-round differential characteristic T (see byte $P_{R,0}$ and $P_{R,3}$), we have a 5-round impossible differential characteristic $\mathbf{D_3}$ $0^8 aa0^8 a0^8 0^8 0^8 \overset{5}{\not\rightarrow} 0^8 s0^8 0^8 0^8 0^8 0^8 0^8$. Note that both S and T occur with a probability of 1 and thus, $\mathbf{D_3}$ will hold with a probability of 0. By adding one round before and after $\mathbf{D_3}$, we can attack 7-round CHAIN-64.

To form such an impossible differential distinguisher, the plaintext XOR difference ΔP must have the form of $\mathbf{S_0^{-1}}(\alpha) \oplus \beta\alpha\alpha\beta\alpha 0^8 \beta 0^8$ and the ciphertext XOR difference ΔC must have the form of $0^8 \Delta_3 \Delta_4 0^8 0^8 \Delta_5 0^8 0^8$. Such forms are obtained by decrypting the input difference of $\mathbf{D_3}$ and encrypting the output difference of $\mathbf{D_3}$.

5.5.2 (a) Attack Procedure

We now can devise the following impossible differential attack procedure to break the full 7-round CHAIN-64 illustrated in Figure 5.9.

1) Choose 2^ϕ structures $\mathscr{S}_i$ (a specific value of ϕ will be given below and for $i = 1,2,\cdots,2^\phi$) where a structure is defined to be a set of 2^{48} plaintexts $P_{i,j}$ (for $j = 1,2,\cdots,2^{48}$) with $P_{L,0}, P_{L,1}, P_{L,2}, P_{L,3}, P_{R,0}$ & $P_{R,2}$) take all the possible 8-bit values and $P_{R,1}$ & $P_{R,3}$ are fixed to a certain value. In a chosen plaintext attack scenario, obtain all the ciphertexts for the 2^{48} plaintexts in each of the 2^ϕ structures. We denote $C_{i,j}$ as the ciphertext for plaintext $P_{i,j}$.

2) For each structure, perform the following sub-steps:

   a) For $1 \leq l \leq 2^{48}$, identify all plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ such that $P_{i,j} \oplus \widehat{P}_{i,l}$ is equal to $\mathbf{S_0^{-1}}(\alpha) \oplus \beta\alpha\alpha\beta\alpha 0^8 \beta 0^8$ and $C_{i,j} \oplus \widehat{C}_{i,l}$ is equal to $0^8 \Delta_3 \Delta_4 0^8 0^8 \Delta_5 0^8 0^8$. Note that $\alpha, \beta, \Delta_3, \Delta_4$ and Δ_5 represent a non-zero byte and they may or may not equal with each other.

   b) Guess a value for $K_{0,1,1}$ & $K_{0,2,0}$ and perform the following sub-steps:

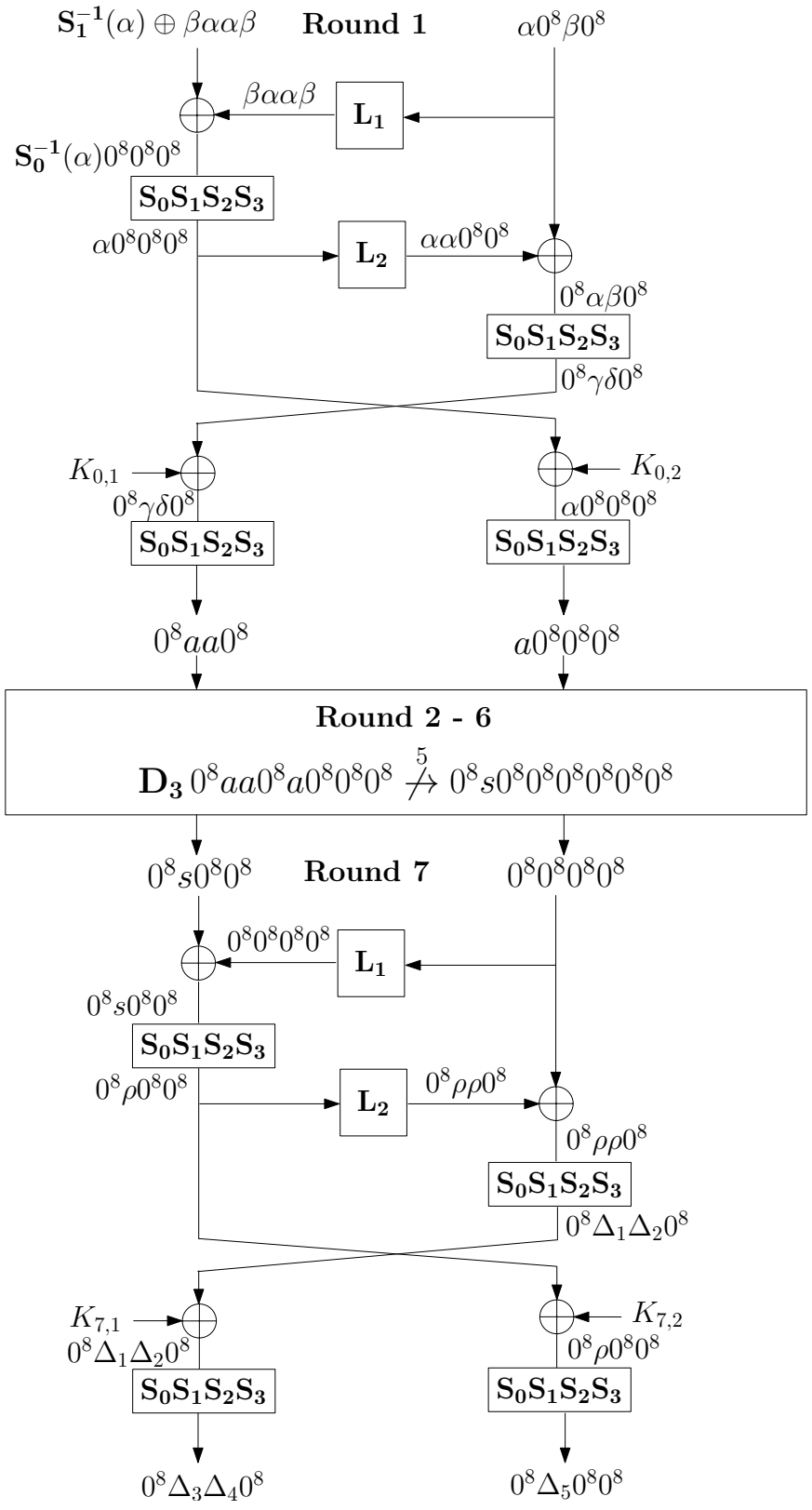


Figure 5.9: The Impossible Differential Attack on 7-Round CHAIN-64

(i) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$. compute $\mathbf{g_1}(P, \widehat{P}) = \mathbf{f_1}(P) \oplus \mathbf{f_1}(\widehat{P}) \oplus \mathbf{f_2}(P) \oplus \mathbf{f_2}(\widehat{P})$ such that $\mathbf{f_1}(X) = \mathbf{S_0}(\mathbf{S_0}(X_{L,0} \oplus X_{R,2} \oplus X_{R,3}) \oplus K_{0,2,0})$ and $\mathbf{f_2}(X) = \mathbf{S_1}(\mathbf{S_1}(\mathbf{S_0}(X_{L,0} \oplus X_{R,2} \oplus X_{R,3}) \oplus \mathbf{S_1}(X_{L,1} \oplus X_{R,0} \oplus X_{R,3}) \oplus X_{R,1}) \oplus K_{0,1,1})$. Discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if $\mathbf{g_1}(P, \widehat{P}) \neq 0$.

(ii) Guess a value for $K_{0,1,2}$ and perform the following sub-steps:

A) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$, compute $\mathbf{f_3}(P) \oplus \mathbf{f_3}(\widehat{P})$ where $\mathbf{f_3}(X) = \mathbf{S_2}(\mathbf{S_2}(\mathbf{S_1}(X_{L,1} \oplus X_{R,0} \oplus X_{R,3}) \oplus \mathbf{S_2}(X_{L,2} \oplus X_{R,0} \oplus X_{R,1}) \oplus X_{R,2}) \oplus K_{0,1,2})$. Then, discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if $\mathbf{f_3}(P) \oplus \mathbf{f_3}(\widehat{P}) \oplus \mathbf{f_1}(P) \oplus \mathbf{f_1}(\widehat{P}) \neq 0$.

B) Guess a value for $K_{6,1,2}$ & $K_{6,2,1}$ and perform the following sub-steps (see Figure 5.10 for details):

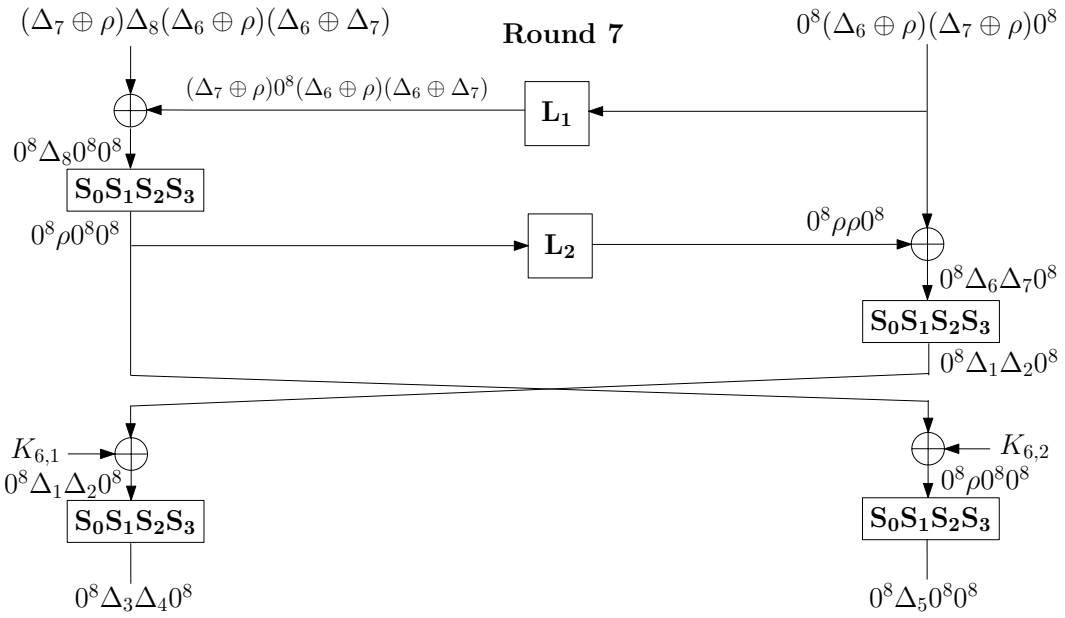


Figure 5.10: 1-Round Decryption of Round 7 of CHAIN-64

I) Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$, compute $\mathbf{g_2}(C, \widehat{C}) = \mathbf{f_4}(C) \oplus \mathbf{f_4}(\widehat{C}) \oplus \mathbf{f_5}(C) \oplus \mathbf{f_5}(\widehat{C})$ where $\mathbf{f_4}(X) = \mathbf{S_2^{-1}}(\mathbf{S_2^{-1}}(X_{L,2}) \oplus K_{6,1,2})$ and $\mathbf{f_5}(X) = \mathbf{S_1^{-1}}(X_{R,1}) \oplus K_{6,2,1}$. Discard those plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ if

$\mathbf{g_2}(C,\widehat{C}) \neq 0$.

II) Guess a value for $K_{6,1,1}$. For each remaining plaintext-ciphertext quartets $(P_{i,j},\ C_{i,j},\widehat{P}_{i,l},\widehat{C}_{i,l})$, compute $\mathbf{f_6}(C) \oplus \mathbf{f_6}(\widehat{C})$ such that $\mathbf{f_6}(X) = \mathbf{S_1^{-1}}(\mathbf{S_1^{-1}}(X_{L,1}) \oplus K_{6,1,1})$. If $\mathbf{f_6}(C) \oplus \mathbf{f_6}(\widehat{C}) = \mathbf{f_5}(C) \oplus \mathbf{f_5}(\widehat{C})$, then the guess of six subkey bytes $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{6,1,2}, K_{6,2,1}$ and $K_{6,1,1})$ is wrong and thus can be filtered out.

Given a sufficient number of chosen plaintext-ciphertext pairs, all the wrong subkey bytes will be discarded. Notice that different impossible characteristics can be utilized to filter out the other wrong subkey bytes for Rounds 1 and 7. Once the round subkeys K_0 and K_6 are successfully recovered, Rounds 1 and 7 of CHAIN-64 can be removed and the same idea can be used to recover the remaining subkeys for Rounds 2 to 6.

5.5.2 (b) Attack Complexity

The attack requires $2^{\phi} \times 2^{48}$ chosen plaintexts. Step 1 has a time complexity of $2^{\phi+48}$ 7-round CHAIN-64 encryptions. For a structure of 2^{48} chosen plaintexts, Step 1 has a memory complexity of $2^{48} \times (6+8)$ bytes $\approx 2^{51.81}$ bytes since two bytes of all plaintexts are fixed to a certain value. The plaintext-ciphertext pairs are stored in a table indexed by the ciphertext.

In Step 2(a), for a structure of 2^{48} chosen plaintexts, a total of $\binom{2^{48}}{2} \approx 2^{95}$ chosen plaintext-ciphertext quartets can be generated. However, the expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{95} \times 2^{-24} \times 2^{-40} \times \frac{100}{256} \approx 2^{29.64}$. Notice that according to the difference distribution table, for the S-box $\mathbf{S_0}$, there are expected 100 out of 256 possible non-zero input differences x such that $\mathbf{S_0}(x) = y$ for a fixed non-zero output difference y. Thus, the memory complexity of Step 2 is $2^{29.64} \times (6+8+6+3)$ bytes $\approx 2^{34.17}$ bytes since five bytes of both ciphertexts are similar. Since there is a 65.36-bit filtering condition in Step 2 for the differences in the plaintext-ciphertext quartet, thus Step 2 has a time complexity of about $2^{29.64}$ memory

accesses.

Next, we analyse Step 2(b) for a structure of $2^{29.64}$ chosen plaintext-ciphertext quartets. Since there exists 2^{16} possible values of $(K_{0,1,1}, K_{0,2,0})$, Step 2(b)(i) requires $2^{29.64} \times 2^{16} \times (23 \text{ XOR } + 12 \text{ S-box lookup})$ operations. Since 1-round CHAIN-64 encryption requires 3 rotations, 6 XOR and 16 S-box table lookup operations, thus we consider the sum of 23 XOR and 12 S-box lookup operations equals $\frac{35}{25}$ of the time complexity of 1-round CHAIN-64 encryption. Step 2(b)(i) has a time complexity of $2^{29.64} \times 2^{16} \times \frac{35/25}{7} \approx 2^{43.32}$ 7-round CHAIN-64 encryptions. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{29.64} \times 2^{-8} = 2^{21.64}$. Step 2(b)(ii)(A) requires 23 XOR and 12 S-box table lookup operations (which should be less than $\frac{35}{25}$ of the complexity of 1-round CHAIN-64 encryption) for each guess of 2^8 possible values of $K_{0,1,2}$, thus it has a time complexity of $2^8 \times 2^{21.64} \times \frac{35/25}{7} \approx 2^{27.32}$ 7-round CHAIN-64 encryptions. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{21.64} \times 2^{-8} = 2^{13.64}$. Subsequently, Step 2(b)(ii)(B)(I) requires 7 XOR and 6 inverse of S-box table lookup operations (which should be less than $\frac{13}{25}$ of the complexity of 1-round CHAIN-64 encryption) for each guess of 2^{16} possible values of $(K_{6,1,2}, K_{6,2,1})$, thus it has a time complexity of $2^{16} \times 2^{13.64} \times \frac{13/25}{7} \approx 2^{25.89}$ 7-round CHAIN-64 encryptions. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{13.64} \times 2^{-8} = 2^{5.64}$. Finally, Step 2(b)(ii)(B)(II) requires 6 XOR and 6 inverse of S-box table lookup operations (which should be less than $\frac{12}{25}$ of the complexity of 1-round CHAIN-64 encryption) for each guess of 2^8 possible values of $K_{6,1,1}$, thus it has a time complexity of $2^8 \times 2^{5.64} \times \frac{12/25}{7} \approx 2^{9.78}$ 7-round CHAIN-64 encryptions.

In total, the impossible attack has a time complexity of approximately $2^{\phi+48} + 2^{\phi} \times (2^{43.32} + 2^{27.32} + 2^{25.89} + 2^{9.78}) \approx 2^{\phi+48.06}$ 7-round CHAIN-64 encryptions and $2^{\phi+29.64}$ memory accesses.

As mentioned in Section 3.3.3 (c), the question on how many memory accesses (table lookups) are equivalent to one CHAIN-64 encryption in terms of time depends closely on the used platform and CHAIN-64 implementation as well as the storage location of the sorted table. In theoretical block cipher cryptanalysis, it is usually assumed by default that a table is stored in an ideal place, RAM say, like an S-box table and it takes an almost constant time to access an entry in a sorted table independent of the number of entries. Thus, an extremely conservative estimate is that 7 memory accesses equal a 7-round CHAIN-64 encryption in terms of time assuming that the **F** function with a round subkey is precomputed in a table and is equivalent to one memory access by neglecting the computational complexity for the key schedule. Thus, one round is equivalent to one memory access. As a consequence, the total time complexity of the impossible differential attack is $2^{\phi+48.06} + \frac{2^{\phi+29.64}}{7} \approx 2^{\phi+48.06}$ 7-round CHAIN-64 encryptions.

The attack succeeds if the expected number of subkey bytes that will not be filtered out after performing the above attack procedure is less than 1 as one out of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{6,1,2}, K_{6,2,1}, K_{6,1,1})$ must be correct. In the beginning of Step 2(b)(ii)(B)(II), the expected remaining chosen plaintext-ciphertext quartets in all the structures is $2^{\phi+5.64}$. Observe that given any guess of $K_{6,1,1}$, the last condition is fulfilled with the probability of 2^{-8} and thus the probability that each of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{6,1,2}, K_{6,2,1}, K_{6,1,1})$ is filtered out using one of $2^{\phi+5.64}$ possible chosen plaintext-ciphertext quartets is 2^{-8}. It follows that the expected number of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{6,1,2}, K_{6,2,1}, K_{6,1,1})$ that will not be filtered out after performing the above attack procedure is $2^{48} \times (1-2^{-8})^{2^{\phi+5.64}}$. Thus, by letting $\phi = 7.42$, only one of the 2^{48} possible values of $(K_{0,1,1}, K_{0,2,0}, K_{0,1,2}, K_{6,1,2}, K_{6,2,1}, K_{6,1,1})$ will remain and it must be the right subkey.

As a conclusion, the impossible differential attack has a time complexity of $2^{\phi+48.06} = 2^{55.48}$ 7-round CHAIN-64 encryption, a memory complexity of $2^{51.81}$ bytes (which is dominated in Step 1) and a data complexity of $2^{\phi+48} = 2^{55.42}$ chosen plaintexts.

5.6 Summary

The CHAIN block cipher has a variable block length, a variable secret key length and a variable number of rounds. It is a byte-oriented block cipher designed by Peyravian and Coppersmith. In this chapter, we have described an r'-round impossible differential characteristic on CHAIN for a variable block length where r denotes the minimum number of rounds to thwart both differential and linear attacks and $r' \in \{r-1, r\}$. Building on such r'-round impossible differential characteristics, we have presented a generic impossible differential attack on $(r'+2)$-round CHAIN cipher. To show the validity of our generic attack, we have presented two impossible differential attacks using two concrete examples, i.e., 7-round CHAIN-64 and 8-round CHAIN-128. This shows that the full CHAIN cipher is not as secure as claimed by the designers. To the best of our knowledge, this is the first known cryptanalytic attack against CHAIN.

CHAPTER 6

GCM REVISITED

The Galois/counter mode is an authenticated encryption scheme recommended by National Institute of Standards and Technology (NIST) and widely used in a number of well-known cryptographic protocols. GCM is an ISO standard. GCM is constructed by combining the counter mode encryption and the message authentication code GMAC to provide both privacy and authenticity. GMAC can be used as a stand-alone message authentication code. In this chapter, we analyse the security of GMAC and GCM with respect to the forgery and distinguishing attacks. More precisely,

1. We generalise the set of weak key classes proposed by Saarinen in FSE 2012 to include all subsets of nonzero keys. Hence, we remove the condition on the smoothness of $2^n - 1$, where n denotes the block size, for the existence of weak key classes.
2. By considering powers of suitable field elements and linearised polynomials, we further exploit some specific weak key classes to present a universal forgery attack on GMAC.
3. By invoking the birthday paradox arguments, we show that a chosen message attack can be used to distinguish GMAC from a random function.
4. To relax the assumptions required in the universal forgery attack, we show that we can utilise the uniqueness of the counter mode encryption to launch a known ciphertext attack against GCM itself when the initial vector IV is restricted to 96 bits.

The first three attack techniques can be applied to other Wegman-Carter polynomial message authentication codes.

6.1 Introduction

6.1.1 Background

GCM for an n-bit block cipher $\mathtt{E}$ is by far the most widely deployed authenticated encryption algorithm designed by McGrew and Viega (2005). GCM employs an encrypt-then-MAC composition (Bellare & Namprempre, 2008) with the added specification that both encryption scheme and MAC use a same secret key K. In essence, GCM uses the counter mode encryption (CTR) as it can be efficiently pipelined in hardware implementation. As for authentication, GCM uses a polynomial hash over a Galois field $GF(2^n)$ based on Wegman-Carter (1981) universal hashing. We refer to this authentication component as GMAG, the partial authentication component, i.e., the polynomial hash, as GHASH and the GCM counter mode encryption as GCTR.

Apart from the exclusive-or operation, GCM is composed of two main operations, namely, block cipher encryption and multiplication in $GF(2^n)$. Theoretically, the block cipher used in GCM can be of any size $n \geq 64$. In practice, however, AES block cipher (NIST, 2001) with a block size of 128-bit is often used. According to the design specification of GCM, the same secret key K is used in all the block cipher encryptions involved in generating the counter values as well as the authentication key H for the authentication polynomial.

GCM was later standardised by NIST (Dworkin, 2007) and ISO (2009). When paired with AES block cipher as the underlying block cipher, this resulting AES-GCM combination became a replacement for dedicated HMAC (Bellare et al., 1996) in popular cryptographic protocols such as SSH (Igoe & Solinas, 2009), IPsec (Law & Solinas, 2007) and TLS (Salter, Rescorla, & Housley, 2009). GCM has been widely adopted due to its efficiency and high performance. Thus, the security of GCM had been extensively examined by the cryptographic community.

6.1.2 Related Work

The class of Wegman-Carter (1981) polynomials MACs had been widely studied by Bernstein (2005b, 2005a), Bierbrauer, Johansson, Kabatianskii, and Smeets (1994), den Boer (1993) and Taylor (1994). Bernstein (2005b, 2005a) proved that the security of such MACs is retained up to $2^{-n/2}$ authenticated messages. More precisely, the security bounds for such MACs indicate that an n-bit tag provides $2^{-n/2}$ security against forgery (Bernstein, 2005b; Sarkar, 2011).

Ferguson (2005) pointed out two inherent weaknesses in GMAC, especially when short tags (that is, only a certain portion of the n-bit output) are considered. The first weakness raises the probability of a successful forgery by exploiting the linear behaviour of the authentication polynomial. The second weakness reveals the authentication key once the attacker manages to create successful forgeries. Nonetheless, these weaknesses do not lead to a violation of the claims of GCM security bounds. In addition, since GCM uses the GHASH function to generate a pseudorandom 128-bit starting value for GCTR (i.e., J_0) when the size of the IV is not 96-bit, Ferguson explained briefly that one can expect a collision on the counter values after processing about 2^{64} blocks of data (due to the birthday paradox arguments). Such collisions can then be exploited to recover the authentication key H, thereby resulting in a loss of all authentication security.

From a different point of view, Joux (2006) showed how an adversary can recover the authentication key H of GMAC with a chosen IV attack. More precisely, given two authentication tags with the same IV, the sum of these two tags will cancel out the term involving the IV and this leads to a simple polynomial equation for the authentication key H. Thus, H can be recovered by solving this polynomial equation. For the earlier draft of the NIST version on GCM (Dworkin, 2006), Joux pointed out that we can easily select two different IVs that yield the same starting value for GCTR. These two different IVs have sizes which are not equal to 96-bit. Observe that a collision of the starting values will leak out the information of the authentication key H too. Besides, Joux also explained the reason why GCM is insecure when an all zero

IV is used. Once H is known, the attacker can easily construct an universal forgery attack by replacing IV with an equivalent IV (i.e., an IV giving the same starting value). We note that the attacks proposed by Joux were only applicable to the early draft of the NIST version of GCM (Dworkin, 2006) and tweaks had been proposed in the final version (Dworkin, 2007) to counter them.

In Crypto 2008, Handschuh and Preneel (2008) carried out a thorough analysis on the security of universal hash function based MAC algorithms against key recovery attacks. They extended the elegant attack proposed by Joux on *modified* GMAC. We refer to this attack as "root attack". However, root attack is impractical due to the requirement of a large number of tag verifications. In addition, the expected number of multiplications performed by the verification oracle is about 2^n. The weaknesses are inherent to polynomial hashs over $GF(2^n)$ and not to the GCTR used in GCM. The authors also presented birthday collision attacks on GMAC that required IV reuse which violated the security assumptions of GCM.

In FSE 2012, Saarinen (2012) presented some nontrivial weak authentication key classes (apart from the weak key $H = 0$ pointed out by the designers) in the case where $n = 128$. They showed that such weak keys resulted in "cycling attacks" on GMAC. Briefly, each of these weak authentication keys H has a small order in the multiplicative group $GF(2^{128})\backslash\{0\}$, say $H^m = 1$ for $m < 2^{128} - 1$. Since $2^{128} - 1$ has 512 divisors, we can construct 512 relations of the form $H^m + 1 = 0$, and for each of these relations, there are m different values of H satisfying it. Consequently, for each divisor m of $2^{128} - 1$, we can define $C_m = \{H : H^m = 1\}$ which consists of m different nonzero elements. A message can now be easily forged by swapping any two blocks which are at a distance of m blocks apart.

More recently, Iwata, Ohashi, and Minematsu (2012) studied the security proofs of GCM in Crypto 2012. They first pointed out that the counter collision lemma used in proving the privacy and authenticity of GCM was invalid. They proved that the probability of a counter collision is in fact greater than that claimed by the designers.

This leads to a flaw in the security proofs of GCM given by the designers and thus the claimed security bounds are not justified. Moreover, Iwata et al. proposed a simple distinguishing attack that invalidates the main part of GCM privacy proofs. However, the success probability of the attack is insignificantly small and therefore, it hardly poses any threat to the security bounds. Nonetheless, they took a step forward to repair the original security proofs of GCM and concluded that the security bounds are larger than previously claimed. Interestingly, they showed that GCM has better security bounds when the length of IV is restricted to 96 bits.

6.1.3 Motivation and Contributions

This chapter seeks to analyse the security of GCM with respect to the forgeability and distinguishability of the authentication component GMAC. Our contributions are four-fold.

First, motivated by the notion of weak key classes proposed by Saarinen (2012), we show in this present work that weak key classes of every size exist. More importantly, we show that weak key classes exist for any block size n, and hence, their existence is independent of the smoothness of the multiplicative group order $2^n - 1$. Furthermore, we falsify the claim by Saarinen that fields of sizes 2^n, where $2^n - 1$ is prime or Sophie Germain prime fields should be used in the GCM construction to minimise the number of weak keys.

Second, by partitioning all the nonzero authentication keys into certain specific weak key classes, we demonstrate how the root attack proposed by Handschuh and Preneel (2008) can be adapted to launch a universal forgery attack on GMAC, a stand-alone MAC. This is more powerful than the root attack which is essentially an existential forgery attack.

Ideally, a secure MAC will behave like a pseudorandom function. For our third contribution, we launch a chosen message attack to demonstrate that GMAC can be distinguished from a random function using $O(2^{n/2})$ tagging queries. We remark

that our results do not invalidate the GCM security bound (Bernstein, 2005b; McGrew & Viega, 2005). Instead, we have shown that the bound given is tight. More interestingly, the above weak key, forgery and distinguishing attacks can be applied to other Wegman-Carter polynomial MACs (Bierbrauer et al., 1994; den Boer, 1993; Taylor, 1994), Sophie Germain Counter Mode (Saarinen, 2012) and Poly1305-AES (Bernstein, 2005a).

The universal forgery attack against GMAC requires an exhorbitant number of verification queries and blocks of data. In fact, the total number of blocks of all the queries for a successful attack is $O(2^n)$. Besides, a drawback of the universal forgery attack (as well as the root attack) is its use of a fixed IV for all its verification queries, a scenario which violates the security model of GCM. In order to reduce the total number of blocks and to avoid a reuse of IVs, we observe that the secret key used to generate the counter values for encryption as well as the starting value J_0 used in the generation of the tag is identical. In the design specification of GCM, i.e., when $n = 128$, the starting value J_0 depends on the size of the IV used. When the size of the IV is not 96-bit, it is generated as the GHASH value with the IV as its input. As commented by Ferguson, this allows us to apply the birthday attack to around $2^{n/2}$ blocks to find a repetition of the counter values, and this in turn leads to the recovery of the authentication key H. As this attack does not apply when IVs with 96 bits are used (since the starting values and counter values will be distinct when distinct IVs are used), a natural question arises: Can we recover H using $O(2^{n/2})$ blocks when GCM is restricted to IVs with 96 bits? Recall that it was shown by Iwata et al. (2012) that the security of GCM is more superior in this case.

For our final contribution in this chapter, we give an affirmative answer to this question. Specifically, we consider the scenario in which the size of the IV is 96-bit (a recommended choice in many standards) against the forgery attack on GCM (not GMAC only). We show that the uniqueness property of the counter values leaks information about the GHASH value, namely, it allows us to filter out the wrong GHASH values. In this way, we can launch a known ciphertext attack using $O(2^{n/2})$ tagging

queries and $O(2^{n/2})$ blocks to recover the correct value of H of GCM. Our results are supported by both experimental and theoretical justifications. We believe that this attack is interesting as it demonstrates that the GCM security bound is tight under the practical assumptions that an IV with 96-bit is used and the reuse of IVs is not permitted for both the tagging and verification queries. Instead of the approach undertaken by existing analysis on the security of GCM which focuses on GMAC alone, we adopt a different approach by taking into account both the counter mode encryption and GMAC in our analysis.

Table 6.1 summarises the complexities of the various attacks in terms of the total number of tagging queries made by the attacker, t, the total number of verification queries made by the attacker, v, the maximum number of blocks of the longest query made by the attacker, l and the total number of blocks in all queries, σ. For concreteness, we consider an AES-paired GCM, that is, $n = 128$.

Organisation. The remainder of the chapter is organised as follows. In Section 6.2, we briefly review the specification of GCM. Section 6.3 describes our generalisation of weak key classes which were proposed by Saarinen (2012). Following this, we present a modification of the existing existential forgery attack to result in a more powerful universal forgery attack on GMAC in Section 6.4. By launching the chosen message attack, Section 6.5 provides a simple distinguishing attack on GMAC using $2^{n/2}$ queries. To reduce the total number of blocks in a forgery attack, we present a known ciphertext attack that exploits the uniqueness of the counter mode to recover the authentication key H of GCM in Section 6.6. Finally, Section 6.7 concludes the chapter.

Table 6.1: The Complexities of the Various Attacks on GMAC

Attack	t	v	l	σ	*Source*
Cycling attack*	1	$O(2^{128}/m)$	$O(m)$	$O(2^{128})$	(Saarinen, 2012)
Root attack$^{\triangle}$	1	$O(2^{128}/m)$	$O(m)$	$O(2^{128})$	(Handschuh & Preneel, 2008)
Universal forgery attack$^{\triangledown}$	1	$O(2^{128}/m)$	$O(m)$	$O(2^{128})$	Section 6.4
Birthday attack for $IV \neq$ 96-bit†	$O(2^{64})$	0	1	$O(2^{64})$	(Ferguson, 2005)
Attack for $IV =$ 96-bit‡	$O(2^{64})$	0	1	$O(2^{64})$	Section 6.6

* $m|(2^{128}-1)$, same IV used in queries, $O(2^{128})$ multiplications
$^{\triangle}$ $m < 2^{128}-1$, same IV used in queries, $O(2^{128})$ multiplications
$^{\triangledown}$ $m < 2^{128}-1$, same IV used in queries, $O(2^{128})$ multiplications
† H is recovered, no IV reuse
‡ H is recovered, no IV reuse, $O(2^{128})$ XOR operations

6.2 The Galois/Counter Mode (GCM)

Dworkin (2007) claimed that for an underlying approved block cipher, the block size is 128 bits and the key size is at least 128 bits. GCM involves two main algorithms, namely authenticated encryption and authenticated decryption. We ignore the authenticated decryption since our attacks only focus on the authenticated encryption algorithm.

Given the selection of an approved block cipher and key, there are three input strings to the authenticated encryption function:

- A plaintext P: $|P| \leq 2^{39} - 256$
- Additional authenticated data A: $|A| \leq 2^{64} - 1$
- Initial vector IV: $1 \leq |IV| \leq 2^{64} - 1$

Note that GCM protects the authenticity of the plaintext and the authenticated data. Besides, GCM also protects the confidentiality of the plaintext while the authenticated data is left in the clear.

Here are the two output strings to the authenticated encryption function:

- A ciphertext C: $|C| = |P|$
- An authentication tag T: $|T| = 128, 120, 112, 104$ or 96

For simplicity, we let $|T| = n$, that is, no truncation, throughout the chapter.

The authenticated encryption algorithm involves two major functions as follows:

1. The GCM counter mode encryption GCTR: the component to provide confidentiality
2. The polynomial hash GHASH: the partial component for authentication

6.2.0 (a) $GCTR_K(P, J_0)$

Given a plaintext P, a starting value J_0 and a secret key K, the ciphertext C is constructed as follows:

- If P is the empty string, then return the empty string as C.
- Write $P = P_1||P_2||\ldots||P_m$, where each $P_i, 1 \leq i \leq m-1$ is an n-bit block and P_m is either a complete block or a nonempty partial block.
- For $i = 1$ to m, compute $J_i = \text{incr}(J_0)$. The function $\text{incr}(\cdot)$ treats the right-most 32 bits of its argument as a nonnegative integer and increments this value modulo 2^{32}. Mathematically, $\text{incr}(F||I)$ is $F||(I+1 \bmod 2^{32})$.
- For $i = 1$ to $m-1$, compute $C_i = P_i \oplus \mathtt{E}_K(J_i)$.
- Compute $C_m = P_m \oplus \text{MSB}_{|P_m|}(\mathtt{E}_K(J_m))$.
- Generate $C = C_1||C_2||\ldots||C_m$.

Figure 6.1 illustrates the GCTR function.

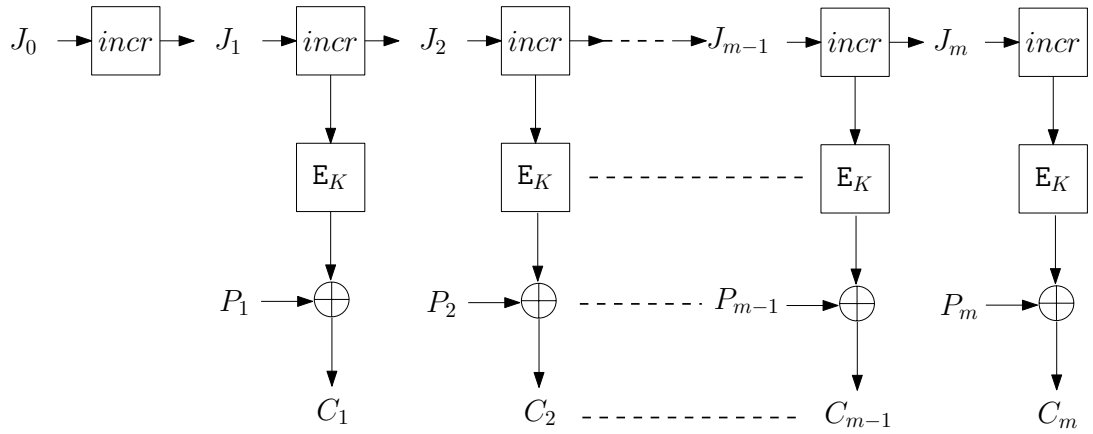


Figure 6.1: The GCTR Function

6.2.0 (b) $GHASH_K(M)$

GHASH is constructed by means of operations in the finite field $\mathrm{GF}(2^n)$. Given a message M and a secret key K, the polynomial hash Y_M is constructed as follows:

- Derive the root authentication key $H = \mathtt{E}_K(0)$.
- Write $M = M_1||M_2||\ldots||M_m$, where each $M_i, 1 \leq i \leq m$ is an n-bit block.
- Define $Y_M = \sum_{i=1}^{m} M_i H^{m-i+1}$.

In addition, GHASH can be implemented using CBC-MAC approach (ISO, 2010). Such approach is illustrated in Figure 6.2.

6.2.0 (c) $GCM\text{-}\mathtt{Enc}_K(IV,P,A)$

Given a plaintext P, an initial vector IV, an optional authenticated data A and a secret key K, the ciphertext C and the authentication tag T are constructed as follows:

- If $|IV| = 96$, then let starting value $J_0 = IV||0^{31}||1$.
- If $|IV| \neq 96$, then let $s = 128\lceil |IV|/128 \rceil - |IV|$ and compute $J_0 = \mathrm{GHASH}_K(IV||0^{s+64}||[|IV|]_{64})$.

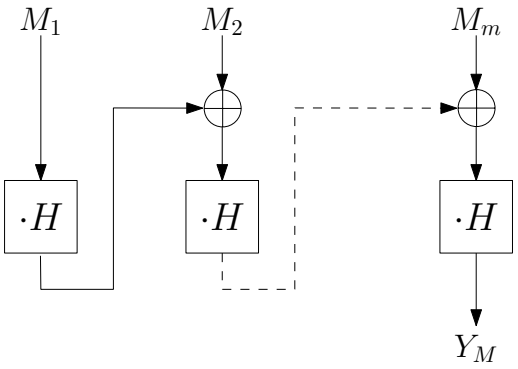


Figure 6.2: The Polynomial Hash GHASH

- Compute $C = \text{GCTR}_K(P, J_0)$.
- Let $u = 128\lceil |C|/128 \rceil - |C|$ and let $v = 128\lceil |A|/128 \rceil - |A|$.
- Compute the tag $T = \text{GHASH}_K(A||0^v||C||0^u||[|A|]_{64}||[|C|]_{64}) + \mathtt{E}_K(J_0)$.
- Return (C, T).

Remark: Suppose we have $M = M_1||M_2||\ldots||M_m$ as the input to GHASH. Define the polynomial $f_M(t) = \sum_{i=1}^{m} M_i t^{m-i+1}$. Then $f_M(t)$ is a polynomial with variable t and degree at most m and $f_M(0) = 0$. Moreover, given two inputs M and M' to GHASH, it is clear that $f_{M+M'}(t) = f_M(t) + f_{M'}(t)$. We observe that the tag T_M corresponding to M can be defined as $T_M = f_M(H) + \mathtt{E}_K(J_0)$ as well. This shows that T_M is a polynomial-evaluation hash with an additional unknown value. For simplicity, we denote the tag generation function by GMAC.

6.3 Weak Keys for GMAC

In the case of GMAC, each secret key K will correspond to an authentication key H (which is not necessarily unique). As such, with respect to the authentication functionality, it suffices to consider weak authentication keys H instead of the weak keys K. In the following, a weak key (see Section 2.1.3 (b) for details) will thus refer to a weak authentication key.

An obvious weak key, as pointed out by the designers, is the key $H = 0$ in which the polynomial $f_M(t)$ evaluates to 0 for any message M. Thus, for any message M, $T_M = \mathrm{E}_K(J_0)$ for a fixed J_0. Henceforth, we assume that $H \neq 0$.

Saarinen (2012) suggested that GCM (or GMAC) may yield more weak keys in the case when $n = 128$ due to the smoothness of 2^{128-1}, the order of the multiplicative group comprising all nonzero elements of $GF(2^{128})$. Specifically, for every divisor m of $2^{128} - 1$, there exists a subgroup of $GF(2^{128})\backslash\{0\}$ of cardinality m. Then every H in this subgroup satisfies $H^m + 1 = 0$. Thus, for all integers $i \geq 1, H^{m+i} + H^i = 0$. Under this relation, it is straightforward to see that swapping two blocks in a message M which are at a distance of m blocks apart will result in the same tag, thereby leading to a successful forgery. Observe that M should have at least $m+1$ blocks for such an attack. Saarinen termed this attack a cycling attack and short cycles result whenever H has a small order m.

Thus, we can define the class $S = \{H : H^m = 1\}$ which, according to Saarinen, qualifies to be a weak key class due to the cycling attack just described. Moreover, since the cardinality of S is exactly m, the probability of any H to be in the class is $\frac{m}{2^{128}-1}$.

Following this line of argument, we now argue that for any $m < 2^n$, there exists a class S of keys satisfying the following conditions:

- For each $H \in S$ and any message M with at least $m+1$ blocks, it is easy to construct a message M' such that $T_M = T_{M'}$.
- S has m elements.

To this end, consider two distinct messages M and M', each with $m+1$ blocks. For a given J_0, the tags T_M and $T_{M'}$ are equal if and only if $f_M(H) + \mathrm{E}_K(J_0) = f_{M'}(H) + \mathrm{E}_K(J_0)$. It follows that $f_M(H) = f_{M'}(H)$ or equivalently, $f_{M+M'}(H) = 0$. Denoting

$f_{M+M'}(t)$ by $g(t)$, we see that M and M' share the same tag if and only if the key H is a root of the polynomial $g(t)$.

This observation motivates us to construct weak key classes as follows. First, partition the set of nonzero elements of $GF(2^n)$ into l disjoint sets, each containing around m elements. We claim that each set can be considered as a weak key class according to the definition given above. Indeed, let $S = \{H_1, H_2, \ldots, H_m\}$ be one of the sets with m elements, where $H_1, H_2, \ldots, H_m \in GF(2^n) \backslash \{0\}$. Given a message M with $m+1$ blocks, our aim is to construct another message M' which shares the same tag as M, that is, $T_M = T_{M'}$, for any key $H_i \in S$. Hence, for each $H_i \in S, H_i$ must be a root of the polynomial $g(t) = f_{M+M'}(t)$.

In view of this, define $g(t) = t\prod_{i=1}^{m}(t-H_i)$. Then for each $j = 1, \ldots, m, g(H_j) = H_j \prod_{i=1}^{m}(H_j - H_i) = 0$ so that H_j is a root of $g(t)$. By expanding $g(t)$, we may express it as $g(t) = \sum_{i=1}^{m+1} c_i t^{m-i+2}$. Note that each c_i is a sum of powers of H_i and are therefore elements of $GF(2^n)$. Given that $M = M_1||M_2||\ldots||M_m||M_{m+1}$, let $M' = M_1 + c_1||\ldots||M_m + c_m||M_{m+1} + c_{m+1}$. Since $f_{M'}(t) = \sum_{i=1}^{m+1}(M_i + c_i)t^{m-i+2} = f_M(t) + g(t), f_{M+M'}(t) = g(t)$. Consequently, for each $H \in S, f_M(H) = f_{M'}(H)$ and this in turn yields $T_M = T_{M'}$.

Since the probability of a key lying in the set S is $\frac{m}{2^n-1}$, we conclude that S is a weak key class satisfying the given two properties. Moreover, since the polynomial $f_{M+M'}(t)$ can have at most m nonzero roots, it is not possible to construct a larger set with more than m elements satisfying the given properties.

In fact, by observing that $\prod_{H,H^m=1}(t-H) = t^m + 1$, we see that our approach generalises the construction of weak key classes proposed by Saarinen (2012). For our construction, we can select nonzero $H_1, \ldots, H_m$ arbitrarily. As a consequence, generalising it this way removes the requirement for $2^n - 1$ to be smooth as proposed by Saarinen (2012), that is, our method applies to any block size. In particular, our generalisation dismisses the suggestion proposed by Saarinen to select fields $GF(2^n)$,

where $2^n - 1$ is a prime or Sophie Germain prime fields to minimise the threat of cycling attacks.

We remark that this idea was already presented by Handschuh and Preneel (2008) in which a forgery attack (which we refer to as the root attack) was given. Here, we model it differently by viewing the keys to lie in a potential weak key class. In this sense, we observe that all the nonzero keys H can be partitioned into distinct weak key classes, thereby leading to the same forgery attack (Handschuh & Preneel, 2008).

It has been brought to our attention that Procter and Cid (2013) have independently made the same observations with regards to general weak key classes as what we have just presented. Specifically, in their paper, they considered the algebraic structure underlying all polynomial hashes and argued that the aforementioned attacks (Ferguson, 2005; Handschuh & Preneel, 2008; Saarinen, 2012) are a consequence of this algebraic property. Inspired by this framework, they showed that almost all subsets of keys can be viewed as weak key classes and exploited different subsets of keys to mount forgery attacks to recover the key H. This slightly differs from the root attack which performs an exhaustive search on the keys in a weak key class that produces a successful forgery.

In this chapter, we exploit the weak key classes in a different way. In general, the root attack is an existential forgery attack in which we can construct a different message M' sharing the same tag value as M. In the next section, we show that there are weak key classes such that a universal forgery attack can be launched.

6.4 Universal Forgery on GMAC

Given an arbitrary message M, the probability of guessing its correct tag is ideally equal to 2^{-n}. In this section, we show that the probability is often much higher, which in turn leads to a universal forgery attack with less than 2^n verification queries. More precisely, we show that the list of possible tags that M can take is less than 2^n.

Now, let M be a message with m blocks, say $M = M_1||M_2||\ldots||M_m$. Let $m' \leq m$ be a positive integer. We consider the following two cases.

Case 1: $m'|(2^n - 1)$

Let $2^n - 1 = lm'$. Since $(H^{m'})^l = H^{2^n-1} = 1$, it follows that $H^{m'}$ is an element of the subgroup of $GF(2^n)\backslash\{0\}$ of order l. In particular, if g is a generator of $GF(2^n)\backslash\{0\}$, then $H^{m'} = g^{im'}$ for some $i = 1,2,\ldots,l$. Moreover, there are exactly m' values of H satisfying $H^{m'} = g^{im'}$ for all $i = 1,\ldots,l$.

Let $f(t) = t^{m'}$. Construct a message $M' = M_1 + 1||M_2||\ldots||M_{m'}||\ldots||M_m$. We have $T_{M'} = f_{M'}(H) + \mathrm{E}_K(J_0) = f_M(H) + H^{m'} + \mathrm{E}_K(J_0) = T_M + H^{m'}$. Since there are l possibilities for $H^{m'}$, we conclude that there are l corresponding possibilities for $T_{M'}$.

Case 2: $m' = 2^k$ **for some** $k < n$

Fix k positions in $\{1,2,\ldots,n\}$. By considering the linear approach in (Ferguson, 2005), we can find a polynomial of the form $f(t) = \sum_{i=0}^{k} c_i t^{2^i}$ such that for all H, $f(H)$ is 0 in each of the k positions. (Here, we assume that all the blocks of the message are arbitrary so that we have $(k+1)n$ variables to begin with). Such a polynomial f is often known as a linearized polynomial. In particular, we can write $f(H) = G\mathbf{h}$, where G is an $n \times n$ binary matrix with the zero rows in the k positions and $\mathbf{h}$ is a vector representing H. Since k bits of $f(H)$ are fixed, it follows easily that $f(H)$ can assume at most 2^{n-k} possible values. Moreover, for each of these values a, the set of H for which $f(H) = a$ can be easily found by solving the matrix equation $G\mathbf{h} = a$. Hence, we conclude that there are exactly 2^{n-k} values of H satisfying each of these equations.

Define $M' = M_1 + c_m||\ldots||M_m + c_1$, where $c_{m'+1} = c_{m'+2} = \ldots = c_m = 0$. Just as in case 1, $T_{M'} = T_M + f(H)$ and hence, there are at most 2^{n-k} possible tags for M'.

In both these cases, notice that we have $T_{M'} = T_M + f(H)$. This allows us to

construct a universal forgery attack on GMAC as follows. Suppose that we wish to obtain the tag for a message $M = M_1||\ldots||M_m$.

- Let k be the largest integer such that $m \geq 2^k$ and let k' be the largest integer not exceeding m with $k'|(2^n - 1)$.
- Define $m' = \max(2^k, k')$.
- If $m' = k'$, construct M' as in case 1 above and set $T = \{g^{m'}, g^{2m'}, \ldots, g^{2^n-1-m'}, 1\}$.
- Otherwise, fix k positions in $\{1, \ldots, n\}$ and let M' as in case 2 above. Set T to comprise all the n-bit strings with 0 in the k positions.
- For a fixed IV, obtain the tag $T_{M'}$ of M' through the tag generation oracle.
- For each $R \in T$, send $T_{M'} + R$ as the tag of M to the verifier.

According to our preceding analysis, we require at most $\frac{2^n-1}{m'}$ tag generations and verifications to forge a valid tag for M. Furthermore, once a forgery is obtained, say $T_M = T_{M'} + R$, the possible values of H is now reduced to the m' roots of the polynomial equation $f(t) = R$. Since these m' roots can be easily determined, the value of H can be recovered by testing each possible root in turn.

Note that Case 1 is more applicable when $2^n - 1$ has many divisors, that is, it is smooth, while Case 2 applies for any block size n. In this attack, the same IV is used for all the verifications.

6.5 Distinguishing Attack on GMAC

In this section, we highlight an interesting observation on the GCM authentication component inspired by the chosen message attack technique, namely, we show that making $O(2^{n/2})$ tag queries will enable us to distinguish the GMAC from a random function with a message of arbitrary length and an IV as inputs and an n-bit output.

We assume that the attacker is given a tag generation oracle and the attacker can request any tag for any messages of his choice. Besides, our distinguishing attack does not require IV reuse. Our distinguishing attack is very simple and makes use of the fact that the underlying block cipher is a bijection.

To be precise, the attacker can request $2^{n/2}$ (or $2^{(n+1)/2}$ respectively) distinct tags for the same message M. Let $T_M = f_M(H) + \mathrm{E}_K(IV)$. If we fix the message $M, f_M(H)$ will be identical in all these tags. Since the block cipher is a bijection, the values of $\mathrm{E}_K(IV)$ will be distinct whenever different IVs are used. As such, the resulting tags will all be distinct.

On the other hand, for a random function which takes in two inputs (a message M and an IV) and outputs an n-bit string, the birthday paradox arguments indicate that a collision will take place after $2^{n/2}$ (or $2^{(n+1)/2}$ respectively) queries (here, M is fixed and we vary IV) with a probability of 0.39 (or 0.63 respectively). In this way, we see that GMAC violates the expected behaviour of a random function since all the tags will be distinct. Hence, after $O(2^{n/2})$ queries, GMAC can be correctly distinguished from a random function with a high probability.

Note that this distinguishing attack is applicable on GMAC and other Wegman-Carter polynomial MACs, but is not applicable on GCM (includes both GCTR and GMAC). This is because in GCM, the attacker does not has any control on the ciphertexts where the attacker cannot obtain distinct tags for the same ciphertext through the GCM encryption oracle.

6.6 Known Ciphertext Attack on GCM

For the universal forgery attack presented as well as the root attack, it is evident that a large number of verification queries and blocks of data are needed. Indeed, as the specification of GCM restricts the number of blocks of a data (i.e., concatenation of ciphertext C and additional authenticated data A) to be within around $2^{32} \cdot 2^{57} = 2^{89}$ blocks, it follows that the total number of verification queries is $O(2^{39})$ which is cer-

tainly too huge to make the attack feasible. This is particularly true with “throttling", where the number of verifications is limited to a small number of queries. In any case, for any block size n, the total number of blocks involved in all the queries for a successful forgery is $O(2^n)$, resulting in around 2^n field multiplications.

More importantly, as we have remarked in Section 6.4, the universal forgery attack requires the reuse of the same IV for all the verifications. This violates the standard security model of GCM (McGrew & Viega, 2005) which prohibits the reuse of an IV for both the tag generation and tag verification oracles.

In order to bypass this issue and to achieve a reduction in the number of verifications and the number of blocks, we attempt to exploit the reuse of the secret K in deriving the starting value J_0 as well as the counter values for the encryption. Throughout this section, we restrict our analysis to the specification of GCM (not GMAC alone) described in Section 6.2.

Recall that the starting value J_0 is defined differently for $|IV| = 96$-bit and $|IV| \neq 96$-bit. In the latter case, J_0 is constructed as the pseudorandom 128-bit output of GHASH with IV as its input. Therefore, two different IVs may produce the same J_0 value and in particular, due to the birthday paradox, we can expect such a collision to occur with $O(2^{n/2})$ queries, with a distinct IV for each query. This attack was already mentioned by Ferguson (2005).

As for a 96-bit IV, J_0 is the concatenation of IV with a 32-bit string $0^{31}1$. Hence, for distinct IVs $IV(1), IV(2), \ldots, IV(k), k \leq 2^{96}$, the bijection property of the encryption operation implies that the following observations hold true.

(1) All the encrypted values $\mathrm{E}_K(J_0(1)), \mathrm{E}_K(J_0(2)), \ldots, \mathrm{E}_K(J_0(k))$ are distinct;

(2) For any $i = 1, \ldots, k$, all the counter values $\mathrm{E}_K(J_1(i)), \mathrm{E}_K(J_2(i)), \ldots, \mathrm{E}_K(J_m(i))$ are distinct (as long as $m < 2^{32}$) and none of these values is equal to $\mathrm{E}_K(J_0(j))$

for any $j = 1, \ldots, k$.

To achieve interoperability, efficiency and simplicity, the use of a 96-bit IV is encouraged in (Dworkin, 2007). As such, we can expect that most of the real world applications of GCM specify a 96-bit IV as a standard. Therefore, we concentrate our analysis on this case for the rest of this section. Specifically, we show that the observations of J_0 and the counter values highlighted above will enable us to recover H using $O(2^{n/2})$ tagging queries and blocks.

To this end, let C be a ciphertext with m blocks and let C' be the input to the polynomial hash GHASH (here, C' will have $m+1$ blocks). For a fixed IV, let $J_0 = IV||0^{31}1$. Then the tag is $T_C = f_{C'}(H) + \mathtt{E}_K(J_0)$. In particular, if $C = 0$ is a one-bit message, it follows from the specification of GHASH that $f_{C'}(H) = H$ or equivalently, $T_C = H + \mathtt{E}_K(J_0)$. By Observation (2), $\mathtt{E}_K(J_0)$ is distinct from $\mathtt{E}_K(J_1), \mathtt{E}_K(J_2), \ldots, \mathtt{E}_K(J_m)$. We claim that $f_{C'}(H) \neq T_C + \mathtt{E}_K(J_i), i = 1, \ldots, m$. Indeed, if equality holds, that is, $f_{C'}(H) = T_C + \mathtt{E}_K(J_i)$ for some $i = 1, \ldots, m$, then $\mathtt{E}_K(J_0) = T_C + f_{C'}(H) = \mathtt{E}_K(J_i)$, contradicting Observation (2). Moreover, given a different starting value J'_0 with counter values $J'_1, J'_2, \ldots, J'_m$, the same argument yields $f_{C'}(H) \neq T_C + \mathtt{E}_K(J'_i)$ for all $i = 1, \ldots, k$.

The preceding arguments enable us to launch a known ciphertext attack to recover H, which we now describe.

- Let F be the set of all n-bit strings.
- Fix a positive integer k which will be specified later.
- Let $IV(1), IV(2), \ldots, IV(2k)$ be $2k$ distinct IVs.
- Let P be a random one-bit message.
- For $i = 1, \ldots, 2k$, query the encryption oracle with inputs $(P, IV(i))$. Since the block cipher is assumed to be random, it can be expected that k of the corre-

sponding one-bit ciphertexts have a value of 0. Record down the tags of these ciphertexts and label them as $T_1, \ldots, T_k$.

- Now, let $IV(1)', \ldots, IV(k)'$ be another k distinct IVs.
- Let P' be a random one-block message.
- For $i = 1, \ldots, k$, query the encryption oracle with inputs $(P', IV(i)')$ to obtain the ciphertexts $C'_1, \ldots, C'_k$.
- Observe that the encrypted counter values $\mathrm{E}_K(J_1(i)')$ for $i = 1, \ldots, k$, can be computed as $\mathrm{E}_K(J_1(i)') = P' + C'_i$.
- For $i, j = 1, \ldots, k$, remove $T_i + \mathrm{E}_K(J_1(j)')$ from F.

According to our procedure, for each $i, j = 1, \ldots, k, T_i + \mathrm{E}_K(J_1(j)') = H + \mathrm{E}_K(J_0(i)) + \mathrm{E}_K(J_1(j)')$ cannot be the correct value for H. Thus, with each new query made, wrong candidates of H can be filtered out.

It now remains to find the number of queries needed to filter out all the wrong candidates of H. Let $A = \{T_1, \ldots, T_k\}$ and $B = \{\mathrm{E}_K(J_1(1))', \ldots, \mathrm{E}_K(J_1(k)')\}$. Note that A and B each contains k distinct random elements. Our task is to find the smallest value of k for which the set $D = \{a + b : a \in A, b \in B\}$ covers almost all of F.

The following lemma is useful to find the smallest value of k.

Lemma 6.1. *Let A and B be two random sets of k n-bit strings. For each n-bit string z, the probability that there exists $a \in A$ and $b \in B$ such that $z = a + b$ is $1 - e^{-k^2/2^n}$.*

Proof. Fix a $z \in GF(2^n)$. Observe that there exists $a \in A$ and $b \in B$ with $a + b = z$ if and only if the sets A and $B + z = \{b + z : b \in B\}$ intersect. The proof of the birthday paradox shows that the probability of selecting a set with k distinct elements is $e^{-k^2/2^{n+1}}$. Hence, it follows that the probability of selecting two sets A and $B + z$, each having k distinct elements is $(e^{-k^2/2^{n+1}})^2 = e^{-k^2/2^n}$. On the other hand, the probability of selecting two

sets A and $B+z$, each having k distinct elements and $A \cap (B+z) = \emptyset$, is $e^{-4k^2/2^{n+1}} = e^{-k^2/2^{n-1}}$ (since it is the same as selecting $2k$ distinct elements). Consequently, the probability for the two sets A and $B+z$ to intersect is $\frac{e^{-k^2/2^n} - e^{-k^2/2^{n-1}}}{e^{-k^2/2^n}} = 1 - e^{-k^2/2^n}$, as desired. □

Since Lemma 6.1 holds for any arbitrary z, it follows that the expected size of $A+B$ is $(1-e^{-k^2/2^n})2^n$. In particular, if $k = 2^{n/2+2}$, the size of $A+B$ is approximately 2^n. Figure 6.3 represents the theoretical proportion of the size of Set D for different values of k. We therefore conclude that if $k \approx 2^{n/2+2}$, H will be recovered. Further, since P and P' are one-block messages, a total of around $3k$ tagging queries are made with a total of $3k$ blocks involved in all the queries.

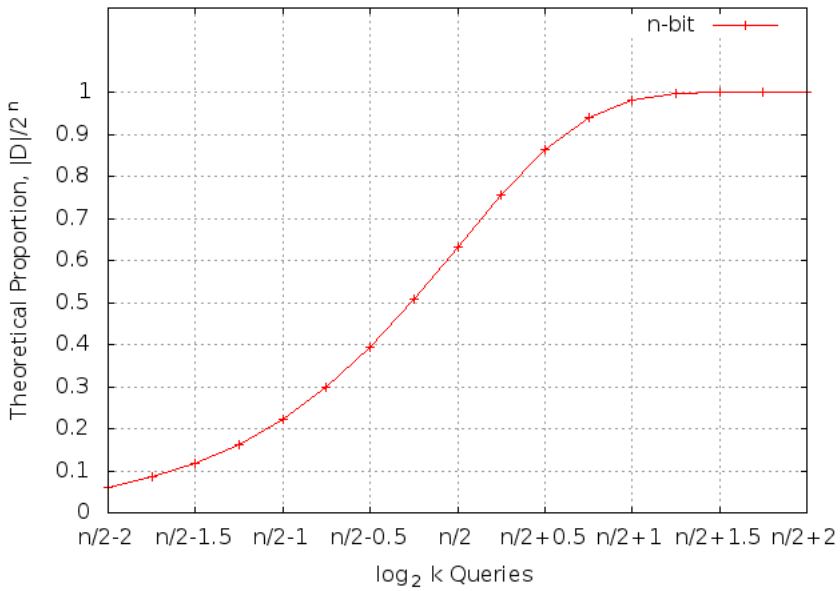


Figure 6.3: The Theoretical Proportion of the Size of Set D for Different Values of k

To confirm our results, we perform some experiments to find the size of $A+B$ for different values of k. We describe the steps of our experiments as follows:

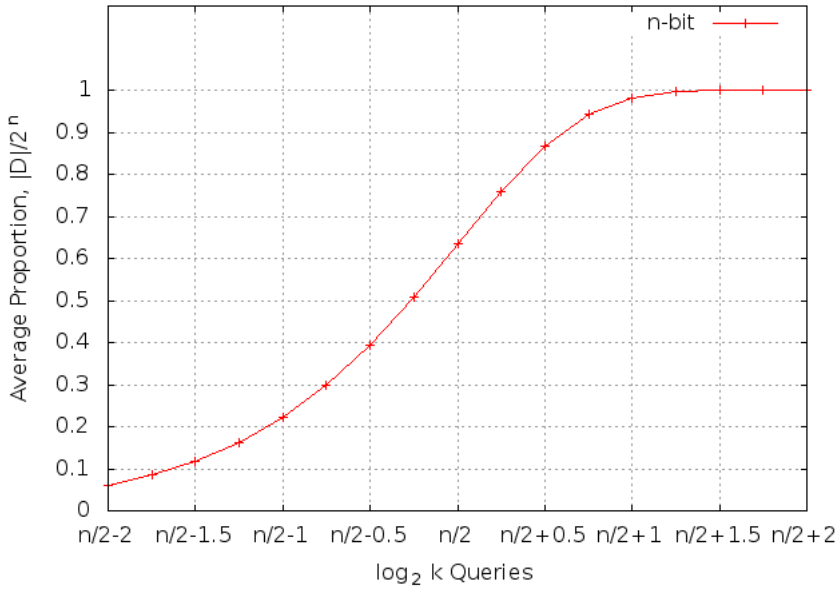


Figure 6.4: The Average Proportion of the Size of Set D for Different Values of k

- Choose k different random n-bit values for Set A.
- Choose k different random n-bit values for Set B. Note that the elements in Set A and Set B may be the same.
- Add each element in Set A with each element in Set B. Record down the sum in Set D.
- Find the size of Set D.
- Compute $|D|/2^n$.

Due to the computational complexity, we run the experiments 1000 times with the parameter $n = 16, 20, 24$ and $k = \{2^{n/2}, 2^{n/2-1.75}, 2^{n/2-1.5}, \ldots, 2^{n/2+1.75}, 2^{n/2+2}\}$. Figure 6.4 presents the average proportion of the size of Set D for different values of k. Observe that our experimental evidence tallies with the theoretical results. In particular, all the n-bit strings are covered with $k = 2^{n/2+2}$ queries.

Observe that in our procedure, $O(2^n)$ XOR operations are performed. This is in contrast to the $O(2^n)$ multiplications required in the root attack and the 2^n block cipher calls required in the brute-force attack. Note that one AES block cipher call requires 160 cycles (Hämäläinen, Alho, Hännikäinen, & Hämäläinen, 2006) while one XOR operation is 80 times faster than one AES block cipher call if we expect each XOR operation requires 2 cycles. Notice that the similar technique had been used in (McGrew, 2012)[1]. McGrew named this technique as impossible plaintext cryptanalysis which can recover information encrypted with the counter mode encryption.

6.7 Summary

In this chapter, we generalised the weak key classes for GMAC and showed that the weak key classes are not limited by the smoothness of $2^n - 1$. This method can be applied to any block size and thus we falsified the suggestion proposed by Saarinen that binary fields $GF(2^n)$ with prime $2^n - 1$ or Sophie Germain prime fields should be used in constructions of this type to minimise the total number of weak keys. Subsequently, we presented an even more powerful universal forgery attack against GMAC by considering powers of suitable field elements and linearized polynomials. To relax the assumptions required in the universal forgery attack against GMAC, we exploited the uniqueness of the counter mode encryption and the bijection property of the underlying block cipher in GCM to recover the authentication key. This attack is applicable to GCM when GCM is used with the recommended 96-bit IV. To the best of our knowledge, this is the first attack when 96-bit IVs are considered. More interestingly, the idea underlying this known ciphertext attack can be extended to launch a distinguishing attack on GMAC and other Wegman-Carter polynomial MACs.

We believe that our analysis via the attacks lead to a better understanding of the security of GMAC and GCM. Similar to the previously proposed attacks, our attacks did not break the security bounds of GCM (McGrew & Viega, 2005; Iwata et al., 2012). Rather, we provided explicit attacks which achieve these bounds in some instances

[1]This work was accepted and presented in FSE 2013, however it was unknown to us that this work was not included in the final proceedings of FSE 2013

with better complexities or are more powerful than the previous attacks.

In order to prevent the above attacks, an obvious countermeasure is to limit the number of blocks that can be processed by a single key to less than $2^{n/2}$ blocks. Besides, the attacks become impractical when n is much greater than 64 due to the large data complexity involved. Thus, a block cipher with a block size of 128 bits (and larger block size to face future challenges) is recommended to pair with GCM. Finally, the user is not recommended to change the key randomly for more than $2^{n/2}$ times as this might lead to the key collision with high probability due to birthday paradox.

CHAPTER 7

PMAC REVISITED

Parallelisable MAC (PMAC) is a part of the OCB mode which is an ISO standard. PMAC was proposed by Black and Rogaway in EUROCRYPT 2002. It is relatively efficient when a parallel environment is possible. This parallelism is achieved via constant multiplications in the underlying finite field. In order to yield a better solution, Rogaway refined PMAC in ASIACRYPT 2004 by using a powering-up construction to generate the constants. This is in contrast to the first design that uses successive words of the gray code to generate the constants. In this chapter, we analyse how some unique characteristics of these constants result in weaknesses of the respective PMAC designs against forgery attacks in different ways. Thus, our analysis highlights some pitfalls that designers should be mindful of when designing schemes which exploit such constants.

7.1 Introduction

MACs can be constructed based on cryptographic hash functions (e.g., HMAC (Bellare et al., 1996)), block ciphers (e.g., CBC-MAC (ISO, 2010), XCBC (Black & Rogaway, 2000), TMAC (Kurosawa & Iwata, 2003) and OMAC (Iwata & Kurosawa, 2003)) or even universal hash functions (e.g., MMH (Halevi & Krawczyk, 1997) and UMAC (Black et al., 1996)). Among these MAC schemes, CBC-MAC and HMAC are the most popular. However, these two schemes share a common characteristic of being inherently sequential where one can only process the i-th message block after all the previous message blocks have been processed.

To counter this bottleneck, Black and Rogaway (2001, 2002) proposed a provably secure and parallelisable MAC (PMAC) scheme based on block ciphers. More precisely, Black and Rogaway proved that PMAC approximates a random function as

long as the underlying n-bit block cipher is a pseudorandom permutation, where n is the size of the block cipher. PMAC was proposed in response to the call of NIST for contributions for the first mode of operation workshop. In ASIACRYPT 2004, Rogaway (2004) proposed the use of a tweakable block cipher for MACs and refined PMAC to yield a more efficient PMAC which uses a powering-up construction for the sequence of constants involved. We refer to the two different versions as PMAC1 (Black & Rogaway, 2001, 2002) and PMAC2 (Rogaway, 2004) throughout this chapter.

For the design of PMACs, the parallelism is achieved at the expense of an extra constant multiplication in the underlying finite field $GF(2^n)$. The distinction between these two variations of PMACs lies in the definition (or generation) of the constants involved. More precisely, PMAC1 uses successive words in a gray code to construct the constants such that two consecutive constants differ in exactly one bit (so the Hamming distance is 1). On the other hand, PMAC2 uses an easier-to-compute sequence of constants comprising $x^1, x^2, \ldots, x^{2^n-1}$ based on the squaring operation. These constants are then multiplied with a certain point $L \in GF(2^n)$ to form the corresponding masks for the blocks. Since it is computationally efficient to multiply a point by x (requiring only a shift and/or an addition), PMAC2 gives rise to a relatively more efficient MAC scheme.

7.1.1 Related Work

Even though PMAC1 had been proposed since 2001, there was no known cryptanalytic result on the PMACs that quantifies the number of message queries (i.e., number of (message, tag) pairs observed) with which a forgery attack can be performed until C. Lee, Kim, Sung, Hong, and Lee (2006) took a first step in analysing the security of PMAC1. They showed how forgery attacks on PMAC1 can be devised, both with truncation (i.e., a tag is truncated to a certain value before being generated) and without truncation (i.e., the length of a tag equals the block size, n).

For PMAC1 without truncation, an attacker first obtains the tags for $2^{n/2+1}$ different messages. Using the birthday paradox arguments, we can expect to find the collision of two tags between two different messages with a probability of 0.63. The attacker can then exploit this collision to find out certain information on the key and thus forge a tag for a new message which does not belong to the earlier set of $2^{n/2+1}$ selected messages.

A similar forgery attack can be carried out in case truncation of the tags is specified. By truncation, we mean that the final tag comprises only the first τ bits of the n-bit output, where $\tau < n$. Once again, an attacker will request the tags of $2^{n/2+1}$ different messages. We can then expect at least $2^{n-\tau}$ of these messages to share the same tag with a probability of 0.63 according to the birthday paradox. However, since collision only applies to the first τ bits (rather than the entire output), additional computations on these $2^{n-\tau}$ messages need to be performed in order to extract useful information to forge other valid tags.

In both these scenarios, we see that the attacker can launch a successful forgery attack with $2^{n/2+1}$ message queries with the success rate of 0.63 due to the birthday paradox. Black and Rogaway (2002) proved that the security bound of PMAC1 was $\frac{(\sigma+1)^2}{2^{n-1}}$ where σ is the total number of message blocks in all q queries made by the attacker. Nandi and Mandal (2008) provided an improved bound of PMAC1 as $\frac{5q\sigma-3.5q^2}{2^n}$. We remark that the attack given by C. Lee et al. does not contradict the security bounds given by Black and Rogaway (2002) and Nandi and Mandal (2008) as the complexity or amount of resources needed by this attack, namely $O(2^{n/2})$ queries, tallies with their bound. Instead, it provides an explicit attack which proves the tightness of their bound. Moreover, note that the key recovery attack on the underlying block cipher proposed by C. Lee et al. (2006) is merely an exhaustive key search attack.

7.1.2 Motivation and Contributions

Given any MAC with an n-bit tag, exploiting the birthday paradox arguments will, with a high probability, lead to a collision of the tags of at least two different messages after $O(2^{n/2})$ (the birthday bound) queries. More importantly, the birthday bound is the security benchmark for all of XCBC, TMAC and OMAC, that is, attacks exist for all of these MACs after obtaining around $2^{n/2}$ (message, tag) pairs. Thus, forgery attacks requiring $O(2^{n/2})$ queries cannot be treated as a weakness of MAC schemes. In fact, in real world practical applications, it is not feasible for the attacker to obtain $O(2^{n/2})$ (message, tag) pairs authenticated with the same key especially if n is much greater than 64.

In order to achieve parallelism, both PMAC1 and PMAC2 employ two different methods to generate the constants. In this chapter, we seek to analyse the effect of the constants on the resilience of the PMAC schemes against forgery attacks. In particular, we present weaknesses of the two PMAC schemes arising from some special properties of the constants which will in turn result in greater forgeability of the schemes. Thus, some extra precautions need to be considered when different methods to generate the constants are used. We also emphasise that our attacks exploit the structural properties of PMAC schemes.

For PMAC1, we show that the use of the gray code to generate the constants renders the scheme *less random* in the following sense: There exists a message M such that the probability of forging a different message having the same tag is higher than 2^{-n}. More significantly, this probability is greater for messages with more message blocks. Observe that in the security proof given by Black and Rogaway (2002, Proof of Lemma 2, Case(D_3, D_5)), the authors claimed that any two random messages, irrespective of their number of message blocks, will have equal tags with a probability of $1/2^n$. However, it follows from our explicit construction that there exist messages sharing the same tag with a much higher probability. This observation motivates us to launch a general birthday attack on PMAC1 which requires fewer than $2^{n/2}$ queries. In particular, our attack works for messages of varying number of blocks, namely, 2^k

blocks with $k > 2$, and exploits an internal collision of the final inputs to a block cipher for two different messages to recover information on the key. We also present a way to extend the attack to yield a more powerful *almost universal forgery attack* using the recovered information on the key.

In addition, we show that three message queries are sufficient to forge a message for PMAC2. The possibility of such a small number of queries for an attack certainly merits our attention as it requires far fewer queries than that suggested by the birthday attack. We thus further explore the feasibility of our attack by investigating the total number of message blocks in our queries. We show that this number of message blocks is influenced by the choice of the irreducible polynomial used in defining the finite field $GF(2^n)$. In particular, the number of message blocks in the messages may go *below* $2^{n/2}$ (breaks the security bounds given by Rogaway (2004) and Minematsu and Matsushima (2007)) for PMAC2 if a *bad* irreducible polynomial is used. Notice that the security bound of PMAC2 proved by Rogaway (2004) is $\frac{5.5\sigma^2}{2^n}$ while Minematsu and Matsushima (2007) gave an improved bound as $\frac{5lq^2}{2^n-2l}$ where l is the maximum number of message blocks of the longest query made by the attacker.

Rogaway (2004) had pointed out the need to check for certain parameters when selecting an irreducible polynomial. More precisely, the designer suggested that one can use discrete-log calculations to help choose and verify that a given choice of parameters[1] (i.e., irreducible polynomial, base and indices) provides a unique representation of the constants. In this chapter, we emphasise the importance of performing these verifications by showing the existence of irreducible polynomials in which the unique representation fails for around $2^{n/2}$ blocks. Further, we provide an explicit forgery attack to demonstrate why repetitions of the constants pose a threat.

These possible security threats against PMACs becomes more important especially when block ciphers (e.g., HIGHT (D. Hong et al., 2006) and PRESENT (Bogdanov et al., 2007)) with a 64-bit block size remain widely used in practice.

[1]Refer to (Rogaway, 2004) for more details

Organisation. The remainder of the chapter is organised as follows. In Section 7.2, we present the specifications of PMAC1 and PMAC2 in detail. We also show our observations on the structure of the gray code. In Section 7.3, we exploit the structure of the gray code to increase the forgery probability of a message and thereby, construct some forgery attacks against PMAC1. We then present a chosen message attack on PMAC2 in Section 7.4 by forging a tag for a specific message that requires only three queries. To study the feasibility of this attack, we investigate the effect of our choice of irreducible polynomials on the total number of message blocks in our queries. In Section 7.5, we discuss the implications of our observations and results. Finally, Section 7.6 concludes the chapter.

7.2 The Parallelisable MAC (PMAC)

In the design of both PMAC1 and PMAC2, operations in the Galois field $GF(2^n)$ are involved. These fields are constructed as the quotient of the ring $GF(2)[x]$ modulo the ideal generated by a fixed monic irreducible polynomial $p_n(x)$ of degree n with binary coefficients. In other words, all elements of $GF(2^n)$ are polynomial expressions of the form $a_0 + a_1x + \ldots + a_{n-1}x^{n-1}$ where $a_0, \ldots, a_{n-1} \in GF(2)$ and all operations are carried out modulo $p_n(x)$. By viewing two binary n-bit strings as elements of $GF(2^n)$, the result of their product (as an n-bit binary string) will therefore depend on the polynomial $p_n(x)$ that defines $GF(2^n)$. For the design of PMAC, Rogaway (2002) chose the lexicographically first polynomial among the irreducible degree n polynomials having a minimum number of coefficients. Here is the list of irreducible polynomials suggested for some parameters of n:

- For $n = 64$, $p_{64}(x) = x^{64} + x^4 + x^3 + x + 1$.
- For $n = 96$, $p_{96}(x) = x^{96} + x^{10} + x^9 + x^6 + 1$.
- For $n = 128$, $p_{128}(x) = x^{128} + x^7 + x^2 + x + 1$.
- For $n = 160$, $p_{160}(x) = x^{160} + x^5 + x^3 + x^2 + 1$.

The general structure of the PMAC schemes consists of three algorithms as depicted in Figure 7.1:

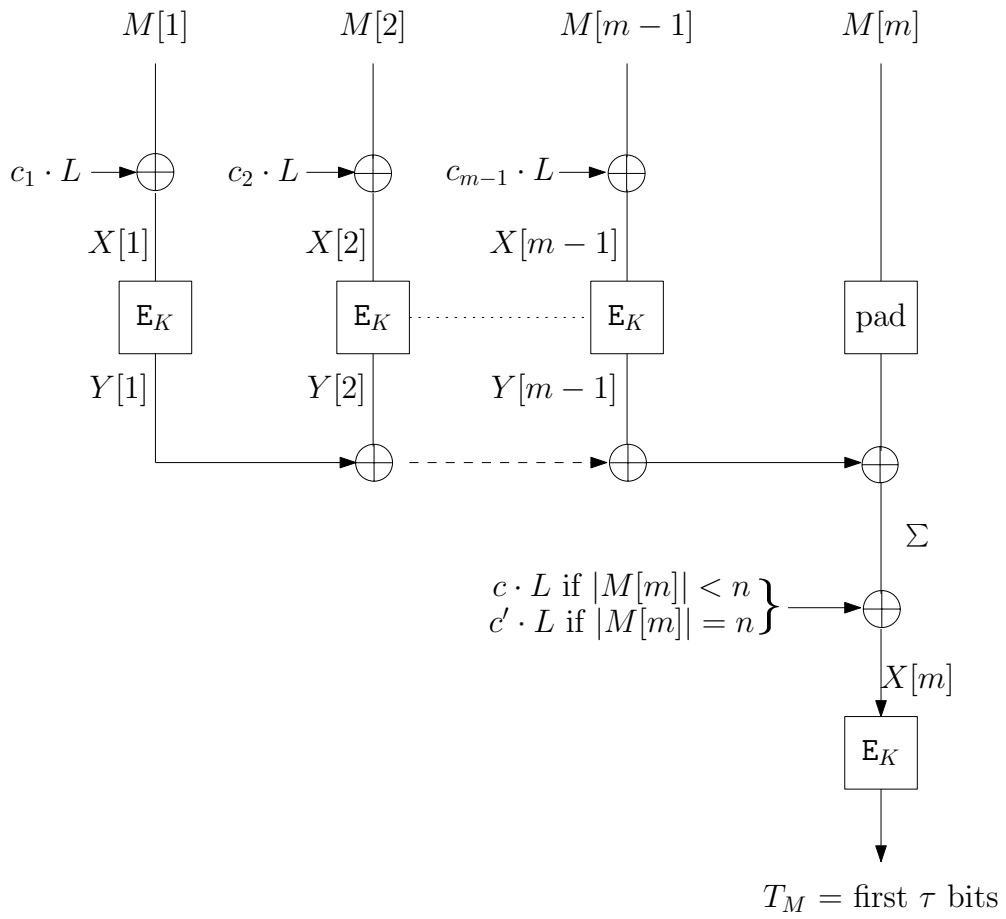


Figure 7.1: Overall Structure of PMAC

1. `KeyGen`:
   - Choose a secret key $K \in \mathscr{K}$ randomly for a block cipher $\mathtt{E}$.
   - Share the secret key K between a sender and a receiver.
   - Both parties compute $L = \mathtt{E}_K(0^n)$, where L denotes the encryption of the message with the value 0 using a block cipher $\mathtt{E}$ with the key K.
2. `Tag`: Given a message M, let $m = \lceil |M|/n \rceil$, where m denotes the number of message blocks in the message M.

- If $|M| > n2^n$, return 0^τ.
- Partition M into $M[1]||\ldots||M[m]$.
- For $i = 1$ to $m-1$, do:
  - Compute $X[i] = M[i] \oplus c_i \cdot L$.
  - Compute $Y[i] = \mathtt{E}_K(X[i])$.
- Compute $\sum = Y[1] \oplus Y[2] \oplus \cdots \oplus Y[m-1] \oplus \mathrm{pad}(M[m])$.
- If $|M[m]| < n$, compute $X[m] = \sum \oplus c \cdot L$.
- If $|M[m]| = n$, compute $X[m] = \sum \oplus c' \cdot L$.
- Output first τ bits of $T_M = \mathtt{E}_K(X[m])$.

3. `Verify` : Given a (M, T_M) pair, do the following:
   - Generate the tag T'_M of M using the `Tag` algorithm.
   - If $T'_M = T_M$, output 1 to indicate the message M as authentic.
   - If $T'_M \neq T_M$, output 0 to indicate the message M as inauthentic.

For simplicity, we let $\tau = n$ in this chapter, that is, we mainly consider the case where no truncation is performed. Here, $c_1, \ldots, c_{m-1}, c$ and c' refer to constants which we are going to describe in detail.

7.2.1 The Generation of Constants for PMAC1

For PMAC1, the sequence of constants is generated by successive nonzero words of the gray code. Specifically, define $c_0 = 0^n$ and $c_1 = 0^{n-1}1$. For $2 \leq i \leq 2^n - 1$, $c_i = c_{i-1} \oplus (0^{n-1}1 \lll \mathrm{ntz}(i))$. We summarise some basic properties of the constants $c_1, \ldots, c_{2^n-1}$ below. For more properties of the gray code, refer to Black and Rogaway (2002).

(i) $c_1, \ldots, c_{2^n-1}$ are all distinct and nonzero.

(ii) The Hamming distance of c_{i-1} and c_i is 1, where $1 \leq i \leq 2^n - 1$.

(iii) As an integer, $c_i < 2i$ for $1 \leq i \leq 2^n - 1$.

An alternative way to construct the words of a gray code of length n is as follows:

- Let $c_0 = 0^n$ and $c_1 = 0^{n-1}1$.
- For $i = 1, 2, \ldots, n-1$, $j = 1, 2, \ldots, 2^i$, define $c_{2^i+j-1} = c_{2^i-j} \oplus (0^{n-1}1 \lll i)$.

As an example, $c_2 = c_1 \oplus 0^{n-2}10 = 0^{n-2}11$, $c_3 = c_0 \oplus 0^{n-2}10 = 0^{n-2}10$, $c_4 = c_3 \oplus 0^{n-3}100 = 0^{n-3}110$, and so on. Hence, we see from this definition that only the least significant bit of c_0 and c_1 can be nonzero while only the 2 least significant bits of c_0, c_1, c_2, c_3 can be nonzero. In general, since the strings $c_{2^i}, c_{2^i+1}, \ldots, c_{2^{i+1}-1}$ are obtained by changing the bit i (let string $a = a_{n-1}a_{n-2}\ldots a_1a_0 \in \{0,1\}^n$, then bit i of a refers to a_i) of $c_{2^i-1}, \ldots, c_0$, respectively, from a 0 to a 1, it follows that for any positive integer i with $i \leq n$, only the i least significant bits of $c_0, c_1, \ldots, c_{2^i-1}$ can be nonzero. Hence, when these strings are viewed as the coefficients of the binary representation of integers, each of these integers is less than 2^i. Since there are exactly 2^i nonnegative integers less than 2^i and all these strings are distinct, we conclude that $c_0, c_1, \ldots, c_{2^i-1}$ comprise all the n-bit binary strings with 0 in all the $n-i$ most significant bits. This leads us to the following theorem.

Theorem 7.1. *For a positive integer $k \leq n$, let $C_k = \{c_0, c_1, \ldots, c_{2^k-1}\}$ and let $C'_k = \{c_{2^k}, c_{2^k+1}, \ldots, c_{2^{k+1}-1}\}$.*

(i) Fix an $a \in C_k$. Then the function f given by $f(y) = y \oplus a$ for all $y \in C_k$ is a bijection from C_k to C_k. In particular, the sequence $c_0 \oplus a, c_1 \oplus a, \ldots, c_{2^k-1} \oplus a$ is a rearrangement of the words in the sequence $c_0, c_1, \ldots, c_{2^k-1}$.

(ii) Fix an $a \in C'_k$. Then the function g given by $g(y) = y \oplus a$ for all $y \in C_k$ is a bijection from C_k to C'_k. In other words, the sequence $c_0 \oplus a, c_1 \oplus a, \ldots, c_{2^k-1} \oplus a$, is a rearrangement of the sequence $c_{2^k}, c_{2^k+1}, \ldots, c_{2^{k+1}-1}$.

Proof. (i) For any $y \in C_k$, the $n-k$ most significant bits of y and a are 0 and thus, the $n-k$ most significant bits of $y \oplus a$ must be 0. It follows from the preceding paragraph that $y \oplus a \in C_k$ so that f is a function from C_k to C_k. Since $f(y) = f(y')$ if and only if $y \oplus a = y' \oplus a$ if and only if $y = y', f$ is injective. Consequently, f must be a bijection from C_k to C_k.
(ii) Let $y \in C_k$. Then $y = 0^{n-k}z$ and $a = 0^{n-k-1}1z'$ for some k-bit strings z and z'. Thus, $y \oplus a = 0^{n-k-1}1z''$, where $z'' = z \oplus z'$. We have $y \oplus a \in C_{k+1}$ but $y \oplus a \notin C_k$. Hence, $y \oplus a \in C'_k$ and g is a function from C_k to C'_k. Similar to the proof in (i), g is a bijection. □

Observe that the constants $c_1, \ldots, c_{m-1}, m \leq 2^n$, are defined as binary strings which are independent of the choice of the irreducible polynomial used in the construction of $GF(2^n)$. However, the binary strings representing the masks $c_1 \cdot L, \ldots, c_{m-1} \cdot L$ may differ according to the polynomial being used. Further details on gray code and $c_i \cdot L$ computations can be found in (Black & Rogaway, 2002).

Finally, we take $c = 0$ and $c' = x^{-1}$ for PMAC1. It is useful to note that these two constants do not depend on m.

7.2.2 The Generation of Constants for PMAC2

The constants for PMAC2 are generated using the powering-up construction. Specifically, define $c_i = x^i$ where $1 \leq i \leq 2^n - 1$. It is easy to see that for each $i, 1 \leq i \leq m-1, X[i] = M[i] \oplus x^i \cdot L$. Thus, if a primitive polynomial $p_n(x)$ is used in constructing $GF(2^n)$, then x will be a generator of the cyclic group of nonzero elements of $GF(2^n)$ so that all the constants $x, \ldots, x^{2^n-1}$ are distinct. Notice that in this case, both the constants c_i and the masks $c_i \cdot L$ may differ when different irreducible polynomials are used in constructing $GF(2^n)$. For PMAC2, $c = x^m(x^2+1)$ and $c' = x^m(x+1)$. Hence, unlike PMAC1, these constants are defined based on the number of message blocks in a message, m.

7.3 On the Security of PMAC1

In this section, we present some weaknesses of PMAC1 against forgery attacks. Specifically, we show in Section 7.3.1 that the structure of the gray code leads to an increased forgery probability for certain messages. Black and Rogaway (2002, Proof of Lemma 2, Case(D_3, D_5)) claimed that given two messages M and M', the probability such that $\sum_M = \sum_{M'}$ is 2^{-n}, where $\sum_M$ and $\sum_{M'}$ represent the output $\sum$ in the tag generation algorithm for M and M', respectively. We show that this is not necessarily the case and in fact, there exist messages M and M' for which the probability is much higher. By exploiting this observation, we present a general birthday attack on PMAC1 in Section 7.3.2. Our birthday attack is more general than the previous one proposed by C. Lee et al. (2006) in the sense that the number of tagging queries to forge a valid (message, tag) pair can be reduced by increasing the number of message blocks for each query. Further, we show that the value of L can be recovered by invoking a few more tagging queries. Finally, in Section 7.3.3, we describe how knowledge of L enables us to launch a more powerful almost universal forgery attack against PMAC1. Such techniques can similarly be applied to the existing attack given by C. Lee et al. (2006).

7.3.1 Constructing Forgeries based on the Structure of the Gray Code

In this section, we seek to demonstrate that the result on the structure of the gray code given in Theorem 7.1 implies that we can construct two distinct messages such that the probability that they share the same tag is higher than 2^{-n}. Our idea is to force the $\sum$ (refer to Figure 7.1) of two distinct messages to the same value. This is similar to the attacks of Gauravaram and Kelsey (2008) on hash functions that use linear checksums where they force the checksum in the attacks to a specific value and then solve the system of linear equations to find the correct blocks for the attacks to be valid. Gauravaram, Kelsey, Knudsen, and Thomsen (2010) then improved their attacks on linear as well as more complicated hash checksums. To this end, we first observe that according to the design specifications of PMAC, the tag generation of a one-block message involves only one encryption since the last message block (which is

the message block itself for a one-block message) is not encrypted. In particular, using the notations in the specification, $X[1] = pad(M[1]) + 0 \cdot L$ in the case where $|M[1]| < n$. This enables us to obtain the encryption of any n-bit message $M_0 \notin \{0^n, 10^{n-1}\}$ as follows:

- Fix an $M_0 \notin \{0^n, 10^{n-1}\}$.
- Let M_0' be such that $\text{pad}(M_0') = M_0$. Clearly, M_0' exists since $M_0 \neq 0^n$ or 10^{n-1}. For example, if $M_0 = a_{k-1}a_{k-2}\ldots a_0 100\ldots 0$, let $M_0' = a_{k-1}a_{k-2}\ldots a_0$ for a positive integer k.
- Request the tag for M_0', $T_{M_0'}$.
- We have $T_{M_0'} = \mathrm{E}_K(\text{pad}(M_0') \oplus 0 \cdot L) = \mathrm{E}_K(M_0)$.

The next lemma is a consequence of Theorem 7.1.

Lemma 7.1. *Let k be a positive integer with $k \leq n-1$. Fix an n-bit message M.*

(i) Let $a \in \{c_1, \ldots, c_{2^k-1}\}$. Consider two 2^k-block messages X and Y defined as follows:

$$X = M||M||\ldots||M||E_K(M),$$

$$Y = (M \oplus a \cdot L)||(M \oplus a \cdot L)||\ldots||(M \oplus a \cdot L)||E_K(M \oplus a \cdot L).$$

We have $T_X = T_Y$, where T_X and T_Y refer to the tags of the messages X and Y, respectively.

(ii) Let $a \in \{c_{2^k}, \ldots, c_{2^{k+1}-1}\}$. Suppose further that $E_K(M \oplus a \cdot L) \notin \{0^n, 10^{n-1}\}$. Let M' be the unique block with less than n bits such that $pad(M') = E_K(M \oplus a \cdot L)$. Consider the following 2^{k+1}-block message:

$$X' = (M \oplus a \cdot L)||\ldots||(M \oplus a \cdot L)||M||\ldots||M||M',$$

where the first $2^k - 1$ blocks of X' are equal to $M \oplus a \cdot L$, the next 2^k blocks of X' are equal to M and the final block is M'. Then the tag of X' is $T_{X'} = L$.

Proof. Define the sets C_k and C'_k as in Theorem 7.1.

(i) We have:

$$\begin{aligned} T_X &= \mathrm{E}_K(\mathrm{E}_K(M \oplus c_1 \cdot L) \oplus \ldots \oplus \mathrm{E}_K(M \oplus c_{2^k-1} \cdot L) \oplus \mathrm{E}_K(M) \oplus c' \cdot L) \\ &= \mathrm{E}_K(\bigoplus_{b \in C_k} \mathrm{E}_K(M \oplus b \cdot L) \oplus c' \cdot L) \end{aligned} \tag{7.1}$$

On the other hand,

$$\begin{aligned} T_Y &= \mathrm{E}_K(\mathrm{E}_K(M \oplus a \cdot L \oplus c_1 \cdot L) \oplus \ldots \oplus \mathrm{E}_K(M \oplus a \cdot L \cdot c_{2^k-1} \cdot L) \oplus \\ &\quad \mathrm{E}_K(M \oplus a \cdot L) \oplus c' \cdot L) \\ &= \mathrm{E}_K(\bigoplus_{b \in C_k} \mathrm{E}_K(M \oplus (a \oplus b) \cdot L) \oplus c' \cdot L) \\ &= \mathrm{E}_K(\bigoplus_{b \in C_k} \mathrm{E}_K(M \oplus b \cdot L) \oplus c' \cdot L) = T_X \end{aligned}$$

Here, the second last equality follows by (i) of Theorem 7.1 since $a \in C_k$.

(ii) In this case, we have:

$$\begin{aligned} T_{X'} &= \mathrm{E}_K(\mathrm{E}_K(M \oplus a \cdot L \oplus c_1 \cdot L) \oplus \ldots \oplus \mathrm{E}_K(M \oplus a \cdot L \oplus c_{2^k-1} \cdot L) \\ &\quad \oplus \mathrm{E}_K(M \oplus c_{2^k} \cdot L) \oplus \ldots \oplus \mathrm{E}_K(M \oplus c_{2^{k+1}-1} \cdot L) \oplus \ \mathrm{pad}(M') \oplus \\ &\quad 0 \cdot L) \\ &= \mathrm{E}_K(\bigoplus_{b \in C_k} \mathrm{E}_K(M \oplus (a \oplus b) \cdot L) \oplus \bigoplus_{b' \in C'_k} \mathrm{E}_K(M \oplus b' \cdot L)) \\ &= \mathrm{E}_K(\bigoplus_{b \in C'_k} \mathrm{E}_K(M \oplus b \cdot L) \oplus \bigoplus_{b' \in C'_k} \mathrm{E}_K(M \oplus b' \cdot L)) \\ &= \mathrm{E}_K(0) = L \end{aligned}$$

Here, $\bigoplus_{b \in C_k} \mathrm{E}_K(M \oplus (a+b) \cdot L) = \bigoplus_{b \in C'_k} \mathrm{E}_K(M \oplus b \cdot L)$ follows from (ii) of Theorem 7.1. □

In view of Lemma 7.1, for a positive integer $k \leq n-1$ and an n-bit message block $M, M \notin \{0^n, 10^{n-1}\}$, denote by $AD(M;2^k)$ the message with 2^k blocks of the form $M||M||\ldots||M||\mathtt{E}_K(M)$, i.e., $AD(M;2^k)$ is a 2^k-block message where the first $2^k - 1$ blocks are equal to M and the final block is the encrypted message $\mathtt{E}_K(M)$. Since $M \notin \{0^n, 10^{n-1}\}, \mathtt{E}_K(M)$ can be obtained through one tagging query and $AD(M;2^k)$ can be constructed. Further, let $T(M;2^k)$ denotes the tag of $AD(M;2^k)$. The next theorem is the main result of this section.

Theorem 7.2. *Fix M to be an n-bit one-block message with $M \notin \{0^n, 10^{n-1}\}$. Let M' be a random n-bit one-block message with $M' \notin \{M, 0^n, 10^{n-1}\}$. Then for any positive integer $k \leq n-1$, the probability that both $T(M;2^k)$ and $T(M';2^k)$ are equal is at least $\frac{2^k-3}{2^n}$.*

Proof. Let $Z = M \oplus M'$. By (i) of Lemma 7.1, if $Z = a \cdot L$ for $a \in \{c_1, \ldots, c_{2^k-1}\}$, then $T(M;2^k) = T(M';2^k)$. Since $Z \notin \{0^n, M, M \oplus 10^{n-1}\}$, the probability that $a \in \{c_1, \ldots, c_{2^k-1}\}$ is at least $\frac{2^k-3}{2^n}$. Consequently, the probability that $T(M;2^k) = T(M';2^k)$ is at least $\frac{2^k-3}{2^n}$. □

When $k > 2$, $\frac{2^k-3}{2^n} \geq \frac{5}{2^n} > 1/2^n$. Thus, Theorem 7.2 shows that there exist distinct messages for which the probability of the messages sharing the same tag is greater than 2^{-n}, contradicting the claim made by Black and Rogaway (2002, Proof of Lemma 2, Case(D_3, D_5)) which essentially asserts that tags of any two random messages collide with a probability of $1/2^n$. In Theorem 7.2, we have shown that this probability can be influenced by the number of message blocks involved.

Recall that Black and Rogaway (2002, Lemma 2) seeks to determine the probability when a certain game involving two distinct messages succeeds. Roughly speaking, it is the sum of probabilities that either the internal blocks of the two messages collide or the final tags collide. Black and Rogaway (2002) termed this probability the MM-collision bound. Suppose that $AD(M;2^k)$ and $AD(M';2^k)$ are the inputs to the MM-collision game. In this case, since all the internal blocks between the two

messages are distinct, the game will succeed with a probability of $\frac{2^k-3}{2^n}$, which is less than the MM-collision bound of $\frac{2^{2k}}{2^n}$ proven by the authors. As such, we believe that the proof of the MM-collision bound (and thus, the security bound) remains valid even though the claims made in cases D_3 and D_5 are ambiguous. Nonetheless, we show that this higher tag collision probability can be exploited in various ways as we illustrate in the following section.

7.3.2 A General Birthday Attack on PMAC1

We begin this section by briefly reviewing the birthday attack proposed by C. Lee et al. (2006). We describe only a simplified version of the attack by selecting one-block messages and considering the tag length, τ, to be n, where n denotes the block size. Note that this attack applies to both PMAC1 and PMAC2. Thus, we refer to the notations used in Section 7.2.

Let A be a set of around $2^{n/2}$ one-block messages such that each message has less than n bits. Let B be a set of around $2^{n/2}$ one-block messages where each message has exactly n bits. According to the birthday paradox arguments, there exist messages $X \in A$ and $Y \in B$ such that $T_X = T_Y$. Hence, $\mathtt{E}_K(\mathrm{pad}(X) \oplus c \cdot L) = \mathtt{E}_K(\mathrm{pad}(Y) \oplus c' \cdot L)$. Notice that $\mathrm{pad}(Y) = Y$. By the bijectivity of $\mathtt{E}_K$, it follows that $\mathrm{pad}(X) \oplus c \cdot L = Y \oplus c' \cdot L$, and L can be found by $L = (\mathrm{pad}(X) \oplus Y)/(c \oplus c')$. With the value of L, for any one-block message M with less than n bits, let $M' = \mathrm{pad}(X) \oplus (c + c') \cdot L$. Then, M and M' will have the same tag and (M', T_M) will be a valid (message,tag) pair.

Observe that $O(2^{n/2})$ tagging queries are required to carry out the above birthday attack. On the other hand, the observation presented in Theorem 7.2 can be exploited to launch a more general birthday attack on PMAC1 that invokes fewer tagging queries at the expense of an increase in the number of message blocks in each query. This is especially effective for the scenario where fewer (message, tag) pairs can be obtained by the attacker. The attack, which is an existential forgery attack, can be described as follows (refer to Figure 7.2):

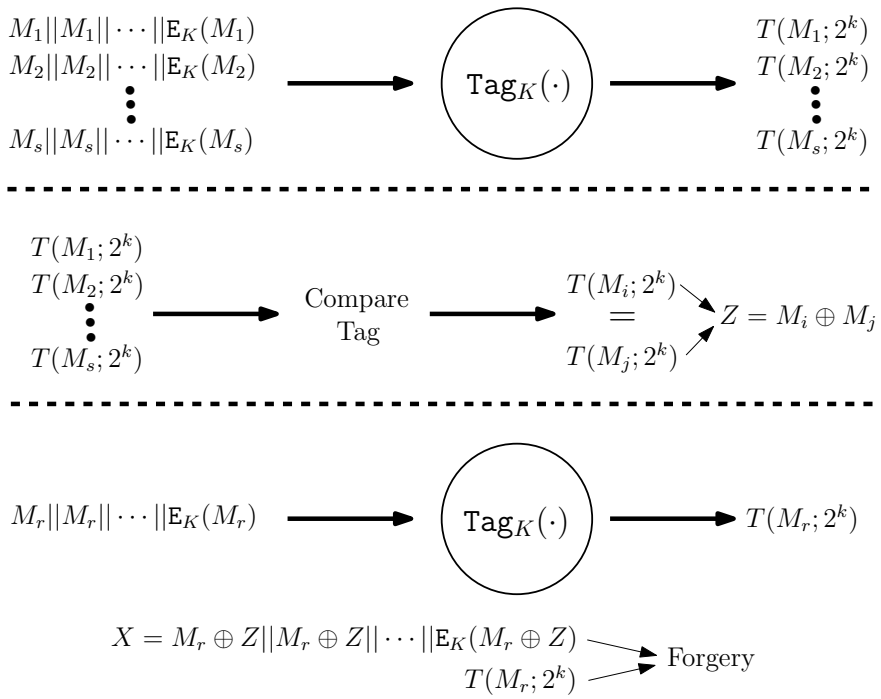


Figure 7.2: Diagram of the General Birthday Attack on PMAC1

1. Fix a positive integer k so that it is feasible to obtain the tag for a message with 2^k blocks.

2. For $s = O(2^{(n-k)/2})$, do:

   - Randomly pick s distinct n-bit one-block messages $M_i \notin \{0^n, 10^{n-1}\}$, $i = 1, \ldots, s$.
   - Obtain the encryption $\mathrm{E}_K(M_i)$ via a tagging query.
   - Construct the messages $AD(M_i; 2^k)$.
   - Query the tagging oracle for the tag of $AD(M_i; 2^k)$ to obtain $T(M_i; 2^k)$.

3. Find i and $j, 1 \leq i < j \leq s$ such that $T(M_i; 2^k) = T(M_j; 2^k)$.

4. Let $Z = M_i \oplus M_j$.

5. Pick an $r, 1 \leq r \leq s, r \notin \{i, j\}$ such that $(M_r \oplus Z) \notin \{0^n, 10^{n-1}, M_1, M_2, \ldots, M_s\}$.

6. Obtain the encryption $\mathrm{E}_K(M_r \oplus Z)$ via the tagging query.

7. Construct the message $X = AD(M_r \oplus Z, 2^k)$.

8. Output the tag $T(M_r; 2^k)$ as the tag of X.

We now explain why our attack works. First, it follows from Lemma 7.1 (i) that there are $O(2^k)$ different messages of the form $AD(M; 2^k)$ such that their tags $T(M; 2^k)$ are identical. As such, all such messages $AD(M; 2^k)$ for different n-bit messages M will have $O(2^{n-k})$ different tags. By the birthday paradox arguments, any set with $s = O(2^{(n-k)/2})$ messages will likely contain two messages (M_i and M_j) producing the same tag, i.e., $T(M_i; 2^k) = T(M_j; 2^k)$. Given that the block cipher is a pseudorandom permutation, we conclude that if $T(M_i; 2^k) = T(M_j; 2^k)$, then it is most likely that $Z = M_i \oplus M_j = a \cdot L$ for some $a \in \{c_1, \ldots, c_{2^k-1}\}$. Consequently, according to Lemma 7.1 (i), the messages $AD(M_r; 2^k)$ and $AD(M_r \oplus Z; 2^k)$ will have the same tags. Moreover, from the way we constructed the message M_r, it is clear that $X = AD(M_r \oplus Z; 2^k)$ had not been queried before. In other words, $(X, T(M_r; 2^k))$ is a valid (message, tag) pair.

So far, we have obtained $Z = a \cdot L$ for $a \in \{c_1, \ldots, c_{2^k-1}\}$. If $a \in C'_{k-1} = \{c_{2^{k-1}}, \ldots, c_{2^k-1}\}$, then Lemma 7.1 (ii) shows that the value of L can be found through the tag of a certain message. Moreover, it follows from the lemma that $a \in C_{k-1}$ (or equivalently, $a \notin C'_{k-1}$) if and only if $T(M; 2^{k-1}) = T(M \oplus Z; 2^{k-1})$ for any n-bit message $M \notin \{0^n, 10^{n-1}\}$. More generally, as long as $a \in C'_j$ for some $j \leq k-1$, the value of L can be determined and we can check whether $a \in C_j$ by checking if $T(M; 2^j) = T(M \oplus Z; 2^j)$. From these observations, we can extend the preceding attack to determine the value of L as follows:

- Pick an n-bit message M so that $M, M \oplus Z$ and $\mathtt{E}_K(M \oplus Z)$ are not in the set $\{0^n, 10^{n-1}\}$.
- Set $j = k-1$.
- While $T(M; 2^j) = T(M \oplus Z; 2^j)$ do: $j := j-1$;
- Let M' be such that $\mathrm{pad}(M') = \mathtt{E}_K(M \oplus Z)$.

- Construct a message X' with 2^{j+1} blocks of the form

$$X' = (M \oplus Z)||\ldots||(M \oplus Z)||M||\ldots||M||M',$$

  i.e., the first $2^j - 1$ blocks of X' are equal to $M \oplus Z$, the next 2^j blocks of X' are equal to M and the final block of X' is M'.

- Obtain the tag of X' through a tagging query and denote it by $T_{X'}$.

- Set $L = T_{X'}$.

Our attack requires $O(2^{(n-k)/2})$ tagging queries to both forge the tag for a new message or to find L. Thus, this attack requires fewer tagging queries than the birthday attacks (C. Lee et al., 2006) which require $O(2^{n/2})$ queries. Consequently, our attack is still effective even if the users can obtain fewer (message,tag) pairs. Nevertheless, if $k \leq n/2$, the total number of blocks (and hence, block cipher calls) is around $O(2^{(n+k)/2}) + O(2^k) = O(2^{(n+k)/2})$.

In order to illustrate the complexity of our attack, we consider a PMAC1 scheme with a 64-bit and 128-bit block cipher. The Table 7.1 gives the complexities of our attack for different values of k.

Table 7.1: Number of Tagging Queries for Different n-Bit Block Sizes and Message Blocks in Each Query

n	log_2 message blocks	# tagging queries
64	16	$O(2^{24})$
64	24	$O(2^{20})$
64	32	$O(2^{16})$
128	32	$O(2^{48})$
128	48	$O(2^{40})$
128	64	$O(2^{32})$

Finally, we remark that all the attacks in this section are general in the sense that they can be applied to messages with varying number of message blocks, namely 2^k blocks, for any positive integer k. More precisely, observe that the birthday attacks (C. Lee et al., 2006) require $q = O(2^{n/2})$ queries in which the maximum number of message blocks of each query is $l = O(1)$. On the other hand, in the worst-case scenario when a "bad irreducible polynomial" is used in the construction of the finite field $GF(2^n)$, our analysis in the next section shows that PMAC2 tags can be forged using $q = O(1)$ queries and $l = O(2^{n/2})$. As such, our attacks on PMAC1 can be viewed as a bridge between these two extremes, since they enable us to forge PMAC1 tags with $l = O(2^k)$ and $q = O(2^{(n-k)/2})$ for any integer k between 1 and $n/2$.

7.3.3 Almost Universal Forgery Attack on PMAC1

In this section, we discuss how knowledge of L leads to an almost universal forgery attack, namely, apart from a few exceptions, we wish to find the tag of any message M without passing M through the tagging oracle. Here, we describe one such procedure. Let M denote a given message. Decompose M into $M = M[1]||\ldots||M[m]$. Suppose that $M[m] \neq x^{-1} \cdot L$ or $10^{n-1} \oplus x^{-1} \cdot L$. We consider two cases.

- **Case 1**: $|M[m]| < n$
  1. Determine $M[m]' = \text{pad}(M[m]) \oplus x^{-1} \cdot L$.
  2. Request the tag of $M' = M[1]||M[2]||\ldots||M[m-1]||M[m]'$. Notice that $M' \neq M$.
  3. Output $T_{M'}$ as the tag of M.

  We see that this attack works since $T_{M'} = \mathrm{E}_K(\bigoplus_{i=1}^{m-1} \mathrm{E}_K(M[i] \oplus c_i \cdot L) \oplus M[m]') = \mathrm{E}_K(\bigoplus_{i=1}^{m-1} \mathrm{E}_K(M[i] \oplus c_i \cdot L) \oplus \text{pad}(M[m]) \oplus x^{-1} \cdot L) = T_M$.

- **Case 2**: $|M[m]| = n$
  1. Determine $M[m]' = M[m] \oplus x^{-1} \cdot L$.
  2. Find $M[m]''$ where $\text{pad}(M[m]'') = M[m]'$. Notice that $|M[m]''| < n$ and $M[m]''$ exists because $M[m]' \neq 0^n, 10^{n-1}$.

3. Request the tag of $M' = M[1]||M[2]||\ldots||M[m-1]||M[m]''$. Observe that $M' \neq M$.

4. Output $T_{M'}$ as the tag of M.

Similar to case 1, this attack works since $T_{M'} = T_M$.

We may use more queries to forge the tag of our exceptional cases but we leave interested readers to work out the details. Notice that this procedure can be applied to the birthday attack proposed by C. Lee et al. (2006) as well since the value of L can be explicitly obtained.

7.4 On the Security of PMAC2

We now turn our attention to the security of PMAC2 against existential forgery attacks. By considering the tag generation algorithm for PMAC, it is apparent that the security of such construction greatly depends on the masks $c_i \cdot L$ used to mask the individual blocks. For instance, if there exist two identical masks for two different blocks of message, an attacker may exploit such a scenario to construct a forgery by swapping the messages in these two different blocks as both messages will generate a same tag (resulting in a collision). This holds true since the addition operation in $GF(2^n)$ (or equivalently, the exclusive-or operation) is commutative.

However, the constants $c_i = x^i, 1 \leq i \leq m-1$ will all be distinct for PMAC2 as long as $m \leq 2^n - 1$. Nonetheless, as the constants used in different messages are the same, we now show how a one-block message can be exploited to construct collisions of messages.

Recall that for PMAC2, the last constants (i.e., c or c') are dependent on the number of message blocks involved. Let M_1 be any one-block message so that $m = 1$. By our construction, the tag for M_1 is $T_{M_1} = \mathrm{E}_K(\mathrm{pad}(M_1) \oplus d \cdot L)$, where $d = x^3 + x$ or $x^2 + x$. Now, suppose that there exists some $k, k \geq 1$ for which $c_k = d$. Define a message $M = M[1]||M[2]||\ldots||M[k]||M[k+1]$, where $M[1], \ldots, M[k-1]$ are arbitrary,

$M[k] = \text{pad}(M_1), M[k+1] = T_{M_1} \oplus R$, for some arbitrary n-bit R. Hence,

$$\begin{aligned} T_M &= \mathtt{E}_K(\bigoplus_{i=1}^{k} \mathtt{E}_K(M[i] \oplus c_i \cdot L) \oplus T_{M_1} \oplus R \oplus c_{k+1} \cdot L) \\ &= \mathtt{E}_K(\bigoplus_{i=1}^{k-1} \mathtt{E}_K(M[i] \oplus c_i \cdot L) \oplus \mathtt{E}_K(M[k] \oplus c_k \cdot L) \oplus T_{M_1} \oplus R \\ &\quad \oplus c_{k+1} \cdot L) \\ &= \mathtt{E}_K(\bigoplus_{i=1}^{k-1} \mathtt{E}_K(M[i] \oplus c_i \cdot L) \oplus R \oplus c_{k+1} \cdot L) \end{aligned} \tag{7.2}$$

Since the constants $c_1, \ldots, c_{2^n-1}$ are all distinct, there must certainly exist some $k, 1 \leq k \leq 2^n - 1$ for which $c_k = d$. Since M_1 is arbitrary in the derivation of T_M in Equation 7.2, we now show that three queries are sufficient to launch a forgery attack on PMAC2.

For PMAC2, $d = x^2 + x$ or $x^3 + x$, depending on the length of the block M_1. Denote by a and b the integers for which $x^a = x^2 + x$ and $x^b = x^3 + x$, respectively. (More details on a and b will be given in the next section.)

- Let M_1 and M_1' be two i-bit messages, where $i = n$ if $a < b$ and $i < n$ if $a > b$. Let $k = \min(a, b)$.
- Query the tagging orcle to obtain the tags T_{M_1} and $T_{M_1'}$ for the messages M_1 and M_1', respectively.
- Fix R to be an n-bit string. Then, construct a message $M = M[1]||\ldots||M[k-1]||\text{pad}(M_1)||M_2$, where $M_2 = T_{M_1} \oplus R$ and $M[1], \ldots, M[k-1]$ are arbitrary.
- Query the tagging oracle to obtain T_M, the tag for M.
- Construct another message $M' = M[1]||\ldots||M[k-1]||\text{pad}(M_1')||M_2'$, where $M_2' = T_{M_1'} \oplus R$.
- Output T_M as the tag for M'.

It is clear that both T_M and $T_{M'}$ are identical since M_1 and M'_1 are not involved in the expression for T_M (and thus $T_{M'}$) in Equation 7.2. Here, 3 queries are required and the total number of message blocks of our queries is $\min(a,b)+3$. Further, we remark that $i \geq 1$ forgeries can be constructed using the above attack with total queries, $q = i+2$ and total number of blocks, $\sigma = i \times \min(a,b)+3$.

7.4.1 Determining the Size of a and b

Evidently, the feasibility of our preceding attack is greatly influenced by the size of a or b. A fixed primitive irreducible polynomial was suggested by Rogaway (2004) for $n = 128$ and the values of a and b were shown to be extremely large for the specified polynomial. Hence, the authors have proposed parameters to guard against our attack.

In this section, we attempt to determine a and b under different irreducible polynomials in order to investigate how large or small they may be. Our analysis shows that the choice of the underlying irreducible polynomial greatly influences the range of these values. As such, care must be taken to check for the values of a and b when new parameters are set for the scheme.

According to our attack on PMAC2, we need the values of a and b for which $x^a = x^2 + x$ and $x^b = x^3 + x$. Now, let u be such that $x^u = x+1$. Since $x^{u+1} = x^2 + x$ and $x^{2u+1} = x(x+1)^2 = x^3 + x$, we have $a = u+1 \mod 2^n - 1$ and $b = 2u+1 \mod 2^n - 1$. As such, we concentrate on finding u satisfying $x^u = x+1$.

In fact, this is the well-known discrete logarithm problem in the field $GF(2^n)$ which can be solved using the index-calculus method with a worst-case sub-exponential running time. For $n \leq 256$, these computations can be carried out within hours.

Recall that a primitive irreducible polynomial needs to be specified in constructing the field $GF(2^n)$. Although different polynomials give rise to isomorphic finite fields $GF(2^n)$, the corresponding value of u (satisfying $x^u = x+1$) may vary

over all the primitive irreducible polynomials of degree n with binary coefficients. To illustrate, we list down the values of u for the various primitive irreducible polynomials of degree 8 in Table 7.2.

Table 7.2: The Values of u for the Various Primitive Irreducible Polynomials of Degree 8

Primitive Irreducible Polynomial	u
$x^8+x^6+x^5+x^2+1$	233
$x^8+x^7+x^2+x+1$	99
$x^8+x^6+x^4+x^3+x^2+x+1$	122
$x^8+x^6+x^5+x+1$	197
$x^8+x^7+x^6+x^3+x^2+x+1$	141
$x^8+x^6+x^3+x^2+1$	23
$x^8+x^4+x^3+x^2+1$	25
$x^8+x^7+x^3+x^2+1$	59
$x^8+x^6+x^5+x^3+1$	16
$x^8+x^7+x^6+x^5+x^4+x^2+1$	134
$x^8+x^7+x^5+x^3+1$	13
$x^8+x^5+x^3+x^2+1$	240
$x^8+x^7+x^6+x+1$	157
$x^8+x^6+x^5+x^4+1$	231
$x^8+x^7+x^6+x^5+x^2+x+1$	115
$x^8+x^5+x^3+x+1$	243

Observe that the value of u ranges from 13 to 243. For $u=13, a=14$ and $b=27$ which implies that $a<2^{n/2}=16$. Our next result shows that for any even n, there exists an irreducible polynomial of degree n such that $u=2^{n/2}$. For more properties on finite fields which are required for the proof, we refer the reader to Lidl and Niederreiter (1984).

Theorem 7.3. *Let n be even and write $n=2n_0$. There exists a monic irreducible polynomial of degree n, denoted by $p(x)$, such that $x^{2^{n_0}}=x+1$.*

Proof. Let $f(x)=x^{2^{n_0}}+x+1$. We first show that $f(x)|(x^{2^n}+x)$. Since $x^{2^{n_0}}=x+1$, 2^{n_0} repeated squarings then yield $x^{2^n}=x^{2^{n_0}}+1=x+1+1=x$. Hence, it follows that $f(x)|(x^{2^n}+x)$. This means that every root of $f(x)$ is an element in the field $GF(2^n)$.

These roots are not elements of $GF(2^{n_0})$ since $\gcd(f(x), x^{2^{n_0}} + x) = 1$. Moreover, as the derivative of $f(x)$ is $1, f(x)$ has no repeated roots, that is, it has 2^{n_0} distinct roots which lie in $GF(2^n) \backslash GF(2^{n_0})$. Now, all subfields of $GF(2^n)$ are of the form $GF(2^d), d|n$. Therefore, $|\bigcup_{d|n, d \neq n/2, n} GF(2^d)| \leq \sum_{d|n, d \neq n/2, n} |2^d < 2^{n/2}$. Hence, there must exist some root β of $f(x)$ which does not lie in any subfield of $GF(2^n)$. Let $p(x)$ be the minimal polynomial of β over $GF(2)$. Then, $p(x)|f(x)$ is an irreducible polynomial of degree n and $\beta^{2^{n_0}} = \beta + 1$. Our result now follows. □

We verify our results using the Magma Algebra Computational System for $n \leq 40$ and find that in fact, there exists a primitive irreducible polynomial such that Theorem 7.3 holds for all these values of n.

Before we conclude this section, we list down the discrete logarithm values of a and b for $n = 64, 96, 128, 160$ and their respective polynomials suggested by the authors in Table 7.3. Once again, our results are obtained within minutes with the aid of the Magma Algebra Computational System.

Table 7.3: Discrete Log Values for Different Irreducible Polynomials

n	$p_n(x)$	$\log_2 a$	$\log_2 b$
64	$p_{64}(x) = x^{64} + x^4 + x^3 + x + 1$	63.071	59.683
96	$p_{96}(x) = x^{96} + x^{10} + x^9 + x^6 + 1$	95.674	95.253
128	$p_{128}(x) = x^{128} + x^7 + x^2 + x + 1$	127.994	127.987
160	$p_{160}(x) = x^{160} + x^5 + x^3 + x^2 + 1$	159.670	159.241

7.5 Discussion

So far, we have presented some possible forgery attacks on both PMAC schemes. Essentially, our attacks hinge on the following characteristics of the sequence of constants $c_1, \ldots, c_{2^n-1}, c, c'$:

- For some m, the constants $c_1, \ldots, \ldots, c_{m-1}, c, c'$ contain a large subset A satisfying the following property: There exist nonzero n-bit strings y such that $A \oplus y = \{z \oplus y : z \in A\} = A$.
- There exists some $i, 1 \leq i \leq m-1$ such that $c_i = c$ or $c_i = c'$.

We have seen that the gray code satisfies the first property with $A = \{c_0, c_1, \ldots, c_{2^k-1}\}$ for any positive integer k and any $y \in A$. As a result, the probability of a successful forgery will increase proportionately with the number of message blocks m. On the other hand, PMAC2 is not as vulnerable to this attack since it is not so straightforward to construct a set A and many strings y satisfying the given property for a given m.

It can be noted that both sets of constants satisfy the second property. However, as pointed out by Black and Rogaway (2002), the value of i for which $c_i = x^{-1}$ will be huge (bigger than 2^{n-1}) irrespective of the irreducible polynomial being used. In view of this, we focus our attention on PMAC2 and we showed that a "bad" choice of primitive irreducible polynomial will lead to a forgery attack with the total number of blocks $\sigma = O(2^{n/2})$. We even gave some counter examples for $GF(2^8)$ in Table 1 to show that there exists some primitive irreducible polynomials of degree 8 that will lead to a forgery attack with total number of blocks fewer than 2^4, hence breaking the security bound of PMAC2.

Apart from selecting a good primitive irreducible polynomial for which the value of i is exhorbitantly large, it may be useful to restrict the number of blocks in the messages that can be authenticated as well. Alternatively, the tagging of a one-block message can be separately treated so as to prevent the possibility of inserting the tag into a long message to cancel out an intermediate block.

Finally, we briefly discuss the case when truncation of the tags is performed (i.e., $\tau < n$) on PMAC2. Following the same attack procedure, we can then construct

a set of $2^{n-\tau}$ messages that contains a message having the same tag as one of our queries. This gives us a probability of $1/2^{n-\tau}$ of forging a valid tag with three queries for PMAC2. If messages in this set can be queried, then the last $n-\tau$ bits of the encrypted output corresponding to our one-block message can be determined, and thus, additional forgeries can be easily constructed.

7.6 Summary

In this chapter, we presented weaknesses of the PMAC schemes arising from two main characteristics of the constants in the construction. These weaknesses allow us to launch forgery attacks on PMAC1 as well as PMAC2. We proposed a general birthday attack on PMAC1 that requires fewer than $O(2^{n/2})$ tagging queries when more message blocks are used for each message query. Similar to the original birthday attack, we can both create forgeries of new messages (existential forgery attack) as well as recover the value of L explicitly. Once L is recovered, we used L to construct an almost universal forgery attack on PMAC1 where we can forge the tags for most of the messages. The attacks we present in this chapter exploit the structural properties of PMAC. Further, our attack on PMAC2 is explicit and independent of the birthday paradox. Even though the number of queries may be surprisingly small (especially in the case of PMAC2), the total number of message blocks is often too large to pose any direct danger to the schemes. Nonetheless, we pointed out that the attacks remain valid according to the specifications given by the authors of the schemes. More importantly, we showed that the total number of blocks of the queries may vary significantly for PMAC2 (which may be as low as $2^{n/2}$), depending on the choice of the irreducible polynomial used in the construction of the underlying finite field. This indicates the danger of fixing an irreducible polynomial randomly or arbitrarily. However, we emphasised that our attacks did not break the security bound of PMACs. Rather, we provided explicit attacks which achieve these bounds in some instances.

CHAPTER 8

WEAKNESSES IN THE KEY SCHEDULE ALGORITHM OF RECENT IMAGE ENCRYPTION SCHEMES USING EXTERNAL KEY

A compulsory condition for the security of an image encryption scheme is that the length of the external secret key should be sufficiently long in terms of bit length. However, the sufficiently long secret key is not a guarantee that the scheme is secure. We emphasise the importance of designing a secure `Key Schedule` algorithm by showing the weakness of the `Key Schedule` algorithms proposed in two recent image encryption schemes, i.e., the X. Wang and Wang scheme and the Norouzi and Mirzakuchaki scheme. Both schemes share a common weakness where the effective space spanned by the subkeys is smaller than the external secret key space. Such smaller subkey space leads to a practical meet-in-the-middle attack against the X. Wang and Wang scheme with the time complexity of $2^{26.322}$ encryptions and a brute-force attack against the Norouzi and Mirzakuchaki scheme with the time complexity of 2^{192} encryptions even though the designers claimed that brute-force attack is applicable against their schemes with the time complexity of 2^{256} encryptions due to a 256-bit external key being used. This chapter serves an important remark to show the relationship between a secure key schedule algorithm with a secure image encryption algorithm.

8.1 Introduction

Even though most of the image encryption schemes do not specifically describe their `Key Schedule` algorithms explicitly as compared to block ciphers, yet an image encryption algorithm can actually also be seen to comprise of three different algorithms, i.e., `Key Schedule`, `E` and `D`. Different with block ciphers, image encryption algorithms (Pareek, Patidar, & Sud, 2003, 2005, 2006; X. Wang & Guo, 2014; Wei, Liao, wo Wong, & Zhou, 2007) are generally constructed based on chaotic maps or chaotic systems. Designers may either treat the initial conditions and/or the control

parameters of chaotic maps or chaotic systems (Gao, Gu, & Chen, 2009; Qi, Chen, Du, Chen, & Yuan, 2005; Yujun, Xingyuan, Mingjun, & Huaguang, 2010) as the secret key directly or use an external secret key to derive the subkeys (e.g., the number of rounds, the initial conditions and/or the control parameters of chaotic maps or chaotic systems) using the `Key Schedule` algorithm. Besides, the `Key Schedule` algorithm of an image encryption algorithm may also treat plain image related information (e.g., the number of pixels in an image or the last pixel value of an image) as part of the input along with the external secret key.

However, many designers of image encryption schemes claim the security of their schemes based on the length of the external secret key. In this chapter, we emphasise the importance in designing a secure `Key Schedule` algorithm as the sufficiently long secret key is not a guarantee that the proposed image encryption scheme is secure. Throughout this thesis, we term the total number of possible subkeys derived from the `Key Schedule` algorithm under the control of an external secret key as the subkey space and the total number of possible external secret keys as the key space.

To highlight the effect of an insecure `Key Schedule` algorithm on the security of the image encryption scheme, we give two concrete examples by showing the weakness of the `Key Schedule` algorithm proposed in two different schemes by X. Wang and Wang (2014) and by Norouzi and Mirzakuchaki (2014). We find out that the effective subkey space of these two schemes is smaller than the key space. The smaller effective space leads to a number of attacks against the aforementioned schemes.

X. Wang and Wang (2014) proposed a novel image encryption algorithm based on dynamic S-boxes constructed by chaotic maps. A 256-bit external secret key is used to derive the initial conditions of two chaotic maps (i.e., the logistic map and the Kent map) with the additional last pixel of a plain image P. Then, the sequence generated by the chaotic maps are used to construct the initial dynamic S-box. During the encryption process, the construction of S-box will change according to the encrypted pixels. Meanwhile, Norouzi and Mirzakuchaki (2014) proposed a fast color image encryption

algorithm based on two hyper-chaotic systems. A 256-bit external secret key is used to generate the initial conditions for these two chaotic systems. The sequences generated by these chaotic systems will then be used for image encryption. Both schemes share some similarities: 1) A 256-bit external secret key is used to derive the subkeys required to encrypt a plain image and 2) Both designers claimed that their schemes are secure against brute-force attack as 256-bit secret key is used.

Even though a 256-bit long secret key K is used in X. Wang and Wang proposed scheme, the effective subkey space is as small as $2^{44.644}$ and this gives rise to a brute-force attack with a practical time complexity of $2^{44.644}$ encryptions only. Furthermore, by using the meet-in-the-middle technique (Diffie & Hellman, 1977), the attacker can even recover the subkeys with the time complexity of $2^{26.322}$ encryptions. On the other hand, the effective key space of the Norouzi and Mirzakuchaki proposed scheme is 2^{64} smaller than the claim made by the designers. Our results show that the designers should pay attention to ensure that their proposed `Key Schedule` algorithm should not produce a smaller effective subkey space as compared to the length of the used external secret key. This is because the recovery of the subkey will enable the attacker to freely decrypt any encrypted image, which leads to the same impact as recovering the external secret key. Table 8.1 summarises the effective key space of the X. Wang and Wang scheme and the Norouzi and Mirzakuchaki scheme.

Table 8.1: The Effective Key Space of Two Image Encryption Schemes

Scheme	Effective Key Space	Source
(X. Wang & Wang, 2014)	$2^{44.644}$	Section 8.3.1
(Norouzi & Mirzakuchaki, 2014)	2^{192}	Section 8.3.2

Organisation. The remainder of this chapter is organised as follows. In the next section, we describe the specification of the `Key Schedule` algorithm proposed by X. Wang and Wang (2014) and by Norouzi and Mirzakuchaki (2014). In Section 8.3,

the security of the proposed Key Schedule algorithms is thoroughly analysed by finding out the effective key space. Section 8.4 concludes the chapter.

8.2 The Key Schedule Algorithm for Different Image Encryption Schemes

Since we only study the security of the image encryption schemes by exploiting weaknesses in their Key Schedule algorithm, we ignore the description of encryption and decryption algorithms. An $m \times n$ image P is denoted as $P(i, j)$, for $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$ and $P(i, j)$ is an 8-bit element, representing an image pixel. For ease of understanding, we adopt a consistent style and notation in describing the Key Schedule algorithm for different proposed image encryption schemes, i.e., X. Wang and Wang (2014) proposed scheme and Norouzi and Mirzakuchaki (2014) proposed scheme. Thus, we let $m = n = 256$. Besides, a 256-bit external secret key K is divided into 32 8-bit subkeys, i.e., $K = (K_1, K_2, \ldots, K_{32})$.

8.2.1 The Key Schedule Algorithm Proposed by X. Wang and Wang

Let $\{i\}$ denotes the decimal part of i (e.g., $\{5.55\} = 0.55$). Given an image P, the last pixel of P is denoted as $pk = P(256, 256)$. The Key Schedule algorithm takes an external secret key K and an image P and performs as follows:

1. Compute $sum = pk \oplus K_1 \oplus K_2 \oplus \ldots \oplus K_{32}$.
2. Let $\mu = 4$ and compute $b = \{(K_{32} + K_1 + sum)/2^8 + 0.01\}$.
3. Compute $d_1 = (K_{26} + K_7 + sum) \bmod 5 + 5$.
4. Compute $d_2 = (K_{27} + K_6 + sum) \bmod 5 + 5$.
5. Compute $x_{1,1} = \{(K_{23} + K_{10} + sum)/2^8 + 0.01\}$.
6. Compute $x_{2,1} = \{(K_{22} + K_{11} + sum)/2^8 + 0.01\}$.
7. Compute $x_{1,2} = \{(K_{21} + K_{12} + sum)/2^8 + 0.01\}$.
8. Compute $x_{2,2} = \{(K_{20} + K_{13} + sum)/2^8 + 0.01\}$.

μ and b are the initial conditions of two chaotic maps, i.e., the logistic map and the Kent map (X. Wang & Wang, 2014) respectively. Note that the subkeys for Round i of the E algorithm are $sk_i = (b, d_i, x_{i,1}, x_{i,2})$ where $i = 1, 2$.

8.2.2 The Key Schedule Algorithm Proposed by Norouzi and Mirzakuchaki

Let $L = 256 \times 256 = 65536$. The Key Schedule algorithm takes an external secret key K and L and performs as follows:

1. Compute $K' = (\sum_{i=1}^{32} K_i) \bmod 256$ where $i \bmod j$ denotes modulo reduction of i by the modulus j.

2. Compute Q_i for $i = 1$ to 32 as shown in Table 8.2.

Table 8.2: Compute Q_1 to Q_{32}

Q_1	=	$K_1 \oplus K_{32} \oplus K'$	Q_{17}	=	$K_{17} \oplus K_{16} \oplus K'$
Q_2	=	$K_2 \oplus K_{31} \oplus K'$	Q_{18}	=	$K_{18} \oplus K_{15} \oplus K'$
Q_3	=	$K_{14} \oplus K_{18} \oplus K'$	Q_{19}	=	$K_{13} \oplus K_{17} \oplus K'$
Q_4	=	$K_{16} \oplus K_{20} \oplus K'$	Q_{20}	=	$K_{15} \oplus K_{19} \oplus K'$
Q_5	=	$K_5 \oplus K_{28} \oplus K'$	Q_{21}	=	$K_{21} \oplus K_{12} \oplus K'$
Q_6	=	$K_6 \oplus K_{27} \oplus K'$	Q_{22}	=	$K_{22} \oplus K_{11} \oplus K'$
Q_7	=	$K_{10} \oplus K_{22} \oplus K'$	Q_{23}	=	$K_9 \oplus K_{21} \oplus K'$
Q_8	=	$K_{12} \oplus K_{24} \oplus K'$	Q_{24}	=	$K_{11} \oplus K_{23} \oplus K'$
Q_9	=	$K_9 \oplus K_{24} \oplus K'$	Q_{25}	=	$K_{25} \oplus K_8 \oplus K'$
Q_{10}	=	$K_{10} \oplus K_{23} \oplus K'$	Q_{26}	=	$K_{26} \oplus K_7 \oplus K'$
Q_{11}	=	$K_6 \oplus K_{26} \oplus K'$	Q_{27}	=	$K_5 \oplus K_{25} \oplus K'$
Q_{12}	=	$K_8 \oplus K_{28} \oplus K'$	Q_{28}	=	$K_7 \oplus K_{27} \oplus K'$
Q_{13}	=	$K_{13} \oplus K_{20} \oplus K'$	Q_{29}	=	$K_{29} \oplus K_4 \oplus K'$
Q_{14}	=	$K_{14} \oplus K_{19} \oplus K'$	Q_{30}	=	$K_{30} \oplus K_3 \oplus K'$
Q_{15}	=	$K_2 \oplus K_{30} \oplus K'$	Q_{31}	=	$K_1 \oplus K_{29} \oplus K'$
Q_{16}	=	$K_4 \oplus K_{32} \oplus K'$	Q_{32}	=	$K_3 \oplus K_{31} \oplus K'$

3. For $i = 1$ to 8, perform as follows:

   - Compute $t_i = 10^9 \times Q_{(i-1)4+1} + 10^6 \times Q_{(i-1)4+2} + 10^3 \times Q_{(i-1)4+3} +$

$Q_{(i-1)4+4}$.

- Compute $x_i = \frac{t_i}{L^2}$.

x_i for $i = 1$ to 8 are treated as the initial conditions of hyper-chaotic systems (Norouzi & Mirzakuchaki, 2014).

8.3 On the Effective Key Space of the `Key Schedule` Algorithm

8.3.1 The X. Wang and Wang Image Encryption Scheme

In this section, instead of recovering the 256-bit secret key K, we aim to recover the subkeys (i.e., sk_1 and sk_2) as the attacker can encrypt and decrypt any image freely with the possession of all the subkeys. From the specification of the `Key Schedule` algorithm given in Section 8.2.1, to encrypt a plain image, the user needs only the information of $b, d_1, d_2, x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}$ derived from the `Key Schedule` algorithm under the control of a 256-bit secret key K.

Lemma 8.1. *There exist* $2^8 = 256$ *possible values of sum.*

Proof. Given that $sum = pk \oplus K_1 \oplus K_2 \oplus \ldots \oplus K_{32}$, K_1, K_2, ..., K_{32} and pk are 8-bit values. Thus, the exclusive-or of all parameters results in 8-bit values, i.e., 0 to 255. Since there exist 2^{256} possible secret keys $K = (K_1, K_2, \ldots, K_{32})$, thus there exist 2^{248} possible secret keys K that will derive the same value of *sum* for every possible known value of pk. □

Lemma 8.2. *There exist* 5 *possible values of* d_i *where* $i = 1, 2$.

Proof. According to Lemma 1, there exist 256 possible values of *sum*. However, $(K_{26} + K_7 + sum) \bmod 5 \in \{0,1,2,3,4\}$ and thus $d_1 = (K_{26} + K_7 + sum) \bmod 5 + 5 \in \{5,6,7,8,9\}$. It is clear that there exist 5 possible values of d_1 only. We performed the simulation in computing the value of d_1 given all possible combinations of

(K_{26}, K_7, sum), as shown in Table 8.3. Similar proof can be given for $d_2 = (K_{27} + K_6 + sum) \bmod 5 + 5$. □

Table 8.3: $\#(K_{26}, K_7, sum)$ **that Results** d_1

d_1	$\#(K_{26}, K_7, sum)$
5	3355444
6	3355443
7	3355443
8	3355443
9	3355443

Lemma 8.3. *There exist* $2^8 = 256$ *possible values of* b.

Proof. Given $b = \{(K_{32} + K_1 + sum)/2^8 + 0.01\}$, there exist 766 possible values of $(K_{32} + K_1 + sum)$ since $(K_{32} + K_1 + sum) \in \{0, 1, 2, \ldots, 3 \cdot 255 = 765\}$. Hence, $(K_{32} + K_1 + sum)/2^8 \in [0, 2.98828125]$ and $(K_{32} + K_1 + sum)/2^8 + 0.01 \in [0.01, 2.99828125]$. Finally, $b = \{(K_{32} + K_1 + sum)/2^8 + 0.01\} \in [0.01, 0.99828125]$. Even though there exist 766 possible values of $(K_{32} + K_1 + sum)$, however after considering the division and $\{\cdot\}$ operations, only 256 possible values of b remain and this is confirmed with the simulation result shown in Table 8.4. □

Lemma 8.4. *There exist* $2^8 = 256$ *possible values of* $x_{1,1}$, $x_{1,2}$, $x_{2,1}$ *and* $x_{2,2}$ *respectively.*

Proof. The proofs are similar as in Lemma 8.3. □

Theorem 8.1. *The effective key space of the Wang & Wang image encryption algorithm is approximately* $2^{44.644}$.

Proof. According to Lemma 8.3 and Lemma 8.4, there exist 256 possible values of sum, b, $x_{1,1}, x_{1,2}, x_{2,1}$ and $x_{2,2}$ respectively. Meanwhile according to Lemma 8.2, there exist 5 possible value of d_1 and d_2 respectively. However, sum is not considered as

Table 8.4: 256 Possible Values of b

0.00218750	0.00609375	0.01000000	0.01390625	0.01781250	0.02171875
0.02562500	0.02953125	0.03343750	0.03734375	0.04125000	0.04515625
0.04906250	0.05296875	0.05687500	0.06078125	0.06468750	0.06859375
0.07250000	0.07640625	0.08031250	0.08421875	0.08812500	0.09203125
0.09593750	0.09984375	0.10375000	0.10765625	0.11156250	0.11546875
0.11937500	0.12328125	0.12718750	0.13109375	0.13500000	0.13890625
0.14281250	0.14671875	0.15062500	0.15453125	0.15843750	0.16234375
0.16625000	0.17015625	0.17406250	0.17796875	0.18187500	0.18578125
0.18968750	0.19359375	0.19750000	0.20140625	0.20531250	0.20921875
0.21312500	0.21703125	0.22093750	0.22484375	0.22875000	0.23265625
0.23656250	0.24046875	0.24437500	0.24828125	0.25218750	0.25609375
0.26000000	0.26390625	0.26781250	0.27171875	0.27562500	0.27953125
0.28343750	0.28734375	0.29125000	0.29515625	0.29906250	0.30296875
0.30687500	0.31078125	0.31468750	0.31859375	0.32250000	0.32640625
0.33031250	0.33421875	0.33812500	0.34203125	0.34593750	0.34984375
0.35375000	0.35765625	0.36156250	0.36546875	0.36937500	0.37328125
0.37718750	0.38109375	0.38500000	0.38890625	0.39281250	0.39671875
0.40062500	0.40453125	0.40843750	0.41234375	0.41625000	0.42015625
0.42406250	0.42796875	0.43187500	0.43578125	0.43968750	0.44359375
0.44750000	0.45140625	0.45531250	0.45921875	0.46312500	0.46703125
0.47093750	0.47484375	0.47875000	0.48265625	0.48656250	0.49046875
0.49437500	0.49828125	0.50218750	0.50609375	0.51000000	0.51390625
0.51781250	0.52171875	0.52562500	0.52953125	0.53343750	0.53734375
0.54125000	0.54515625	0.54906250	0.55296875	0.55687500	0.56078125
0.56468750	0.56859375	0.57250000	0.57640625	0.58031250	0.58421875
0.58812500	0.59203125	0.59593750	0.59984375	0.60375000	0.60765625
0.61156250	0.61546875	0.61937500	0.62328125	0.62718750	0.63109375
0.63500000	0.63890625	0.64281250	0.64671875	0.65062500	0.65453125
0.65843750	0.66234375	0.66625000	0.67015625	0.67406250	0.67796875
0.68187500	0.68578125	0.68968750	0.69359375	0.69750000	0.70140625
0.70531250	0.70921875	0.71312500	0.71703125	0.72093750	0.72484375
0.72875000	0.73265625	0.73656250	0.74046875	0.74437500	0.74828125
0.75218750	0.75609375	0.76000000	0.76390625	0.76781250	0.77171875
0.77562500	0.77953125	0.78343750	0.78734375	0.79125000	0.79515625
0.79906250	0.80296875	0.80687500	0.81078125	0.81468750	0.81859375
0.82250000	0.82640625	0.83031250	0.83421875	0.83812500	0.84203125
0.84593750	0.84984375	0.85375000	0.85765625	0.86156250	0.86546875
0.86937500	0.87328125	0.87718750	0.88109375	0.88500000	0.88890625
0.89281250	0.89671875	0.90062500	0.90453125	0.90843750	0.91234375
0.91625000	0.92015625	0.92406250	0.92796875	0.93187500	0.93578125
0.93968750	0.94359375	0.94750000	0.95140625	0.95531250	0.95921875
0.96312500	0.96703125	0.97093750	0.97484375	0.97875000	0.98265625
0.98656250	0.99046875	0.99437500	0.99828125		

part of the subkeys that is needed to encrypt a plain image and the same value of b is used in both Round 1 and Round 2. Thus, the number of all possible subkeys is $256^5 \times 5^2 = 27487790694400 \approx 2^{44.644}$. □

A brute-force attack is applicable on the X. Wang and Wang scheme with the time complexity of $2^{44.644}$ encryptions. To further improve the time complexity of the attack, the attacker can utilise the meet-in-the-middle attack technique (Diffie & Hellman, 1977) (see Figure 8.1 for a graphical illustration) due to the small number of rounds as follows.

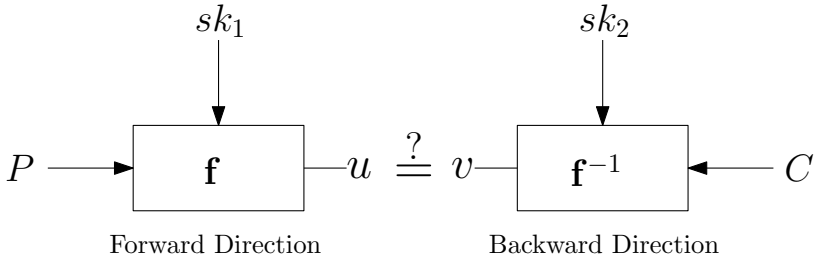


Figure 8.1: The MITM Attack on X. Wang and Wang Image Encryption Scheme

1. Given a pair of plain image and encrypted image (P,C). C is generated by iterating P to a same round function f for two times under the control of subkeys sk_1 and sk_2, i.e., $C = \mathbf{f}_{sk_2}(\mathbf{f}_{sk_1}(P))$.

2. Guess 256 possible values of b, do:

   a) Guess $5 \times 256 \times 256 = 2^{18.322}$ possible values of $(d_1, x_{1,1}, x_{1,2})$ and compute $u = \mathbf{f}_{sk_1}(P)$ where $\mathbf{f}_j(i)$ denotes the one round encryption of input i under the control of a subkey j. Store the result u and the subkey in a memory lookup hash table. Note that the memory complexity is $O(2^{18.322})$.

   b) Guess $5 \times 256 \times 256 = 2^{18.322}$ possible values of $(d_2, x_{2,1}, x_{2,2})$ and compute $v = \mathbf{f}^{-1}{}_{sk_2}(C)$ where $\mathbf{f}^{-1}{}_j(i)$ denotes the one round decryption of input i under the control of a subkey j. Check whether v matches any possible

value of u using the stored memory lookup table. If $v = u$, then (sk_1, sk_2) could be the right subkey since the probability of $v = u$ under the control of a wrong key is negligible.

After performing the above attack, the expected number of possible subkeys remaining is less than 256, thus a brute-force attack can be carried out easily using one additional pair of (P, C) to filter out all the remaining wrong subkeys excepts the right subkey. The meet-in-the-middle attack has a time complexity of around $256 \times 2 \times 2^{18.322}$ one round encryptions $\approx 2^{26.322}$ full encryptions assuming the memory write/access operations are negligible.

Such a practical attack indicates that the X. Wang and Wang image encryption scheme is not suitable for any security application.

8.3.2 The Norouzi and Mirzakuchaki Image Encryption Scheme

From the specification of `Key Schedule` algorithm given in Section 8.2.2, to encrypt a plain image, the user needs only the information of $Q_1, Q_2, \ldots, Q_{32}$ derived from the `Key Schedule` algorithm under the control of a 256-bit external secret key K to generate the initial conditions (i.e., x_i for $i = 1$ to 8) of hyper-chaotic systems since L is known. Since each Q_i is of 8 bits, a naive brute-force attack has the time complexity of 2^{256} encryptions, which is similar as the claim made by Norouzi and Mirzakuchaki. We show that the effective key space of Q_i for $i = 1$ to 32 is actually 2^{64} times smaller.

Lemma 8.5. *If $Q_i \oplus Q_j = Q_k \oplus Q_l$ where $i \neq j \neq k \neq l$, there exists 2^{24} candidates of (Q_i, Q_j, Q_k, Q_l) that fulfill this equation.*

Proof. There are total 2^{32} candidates of (Q_i, Q_j, Q_k, Q_l) since each Q is of 8 bits. The probability of (Q_i, Q_j, Q_k, Q_l) to fulfill the condition of $Q_i \oplus Q_j = Q_k \oplus Q_l$ is 2^{-8} since it is an 8-bit equation. Thus, 2^{24} out of 2^{32} candidates will remain. Lemma 8.5 is verified through a computer simulation too. □

Theorem 8.2. *The effective key space of the Norouzi & Mirzakuchaki image encryption algorithm is* 2^{192}.

Proof. From Table 8.2, we know that

$$\begin{aligned}
Q_3 \oplus Q_{14} &= Q_{18} \oplus Q_{20}, \\
Q_7 \oplus Q_{10} &= Q_{22} \oplus Q_{24}, \\
Q_{11} \oplus Q_6 &= Q_{26} \oplus Q_{28}, \\
Q_{15} \oplus Q_2 &= Q_{30} \oplus Q_{32}, \\
Q_1 \oplus Q_{16} &= Q_{29} \oplus Q_{31}, \\
Q_5 \oplus Q_{12} &= Q_{25} \oplus Q_{27}, \\
Q_9 \oplus Q_8 &= Q_{21} \oplus Q_{23}, \\
Q_{13} \oplus Q_4 &= Q_{17} \oplus Q_{19}.
\end{aligned}$$

According to Lemma 8.5, thus there exist 2^{24} candidates of $(Q_3, Q_{14}, Q_{18}, Q_{20})$, $(Q_7, Q_{10}, Q_{22}, Q_{24})$, $(Q_{11}, Q_6, Q_{26}, Q_{28})$, $(Q_{15}, Q_2, Q_{30}, Q_{32})$, $(Q_1, Q_{16}, Q_{29}, Q_{31})$, $(Q_5, Q_{12}, Q_{25}, Q_{27})$, $(Q_9, Q_8, Q_{21}, Q_{23})$ and $(Q_{13}, Q_4, Q_{17}, Q_{19})$ that fulfill the above equations. By rearranging these candidates, we get the following list of candidates

$$\begin{aligned}
&(Q_1, Q_{16}, Q_{29}, Q_{31}), \\
&(Q_2, Q_{15}, Q_{30}, Q_{32}), \\
&(Q_3, Q_{14}, Q_{18}, Q_{20}), \\
&(Q_4, Q_{13}, Q_{17}, Q_{19}), \\
&(Q_5, Q_{12}, Q_{25}, Q_{27}), \\
&(Q_6, Q_{11}, Q_{26}, Q_{28}), \\
&(Q_7, Q_{10}, Q_{22}, Q_{24}), \\
&(Q_8, Q_9, Q_{21}, Q_{23}).
\end{aligned}$$

Since each Q_i for $i = 1$ to 32 appears once in the above list of candidates, thus the total possible candidates of Q_i for $i = 1$ to 32 is $(2^{24})^8 = 2^{192}$. By knowing Q_i for $i = 1$ to

32, one can deduce t_j and x_j for $j = 1$ to 8 with the known value of L. With the values of x_j, one can encrypt any plain image and decrypt any encrypted image. □

Finally, a brute-force attack can be launched against the Norouzi and Mirzakuchaki image encryption algorithm with the time complexity of 2^{192} encryption and one pair of plain image and encrypted image. An attack with the time complexity of less than 2^{256} also indicates that the underlying `Key Schedule` algorithm is not as secure as designed.

8.4 Summary

Designers of image encryption algorithms should pay extra care in designing the `Key Schedule` algorithm as it may affect the effective key space of their constructed image encryption algorithm. We have analysed the `Key Schedule` algorithm proposed by X. Wang and Wang (2014) and by Norouzi and Mirzakuchaki (2014). The effective key space of both schemes is far smaller than the designers' claim. To make it worse, the key space of the X. Wang and Wang scheme is around $2^{44.644}$ (instead of the claimed 2^{256}) and such small key space gives rise to a practical meet-in-the-middle attack with the time complexity of $2^{26.322}$ encryptions. Even though the effective key space of the Norouzi and Mirzakuchaki scheme is 2^{192} (instead of the claimed 2^{256}) which is still too large to make such brute-force attacks practical, the shrinking of the key space after going through the `Key Schedule` algorithm indicates the weakness of their proposed `Key Schedule` algorithm. One of the possible weaknesses learnt from the above attacks is the smaller output domain of subkeys leads to a smaller entropy. As a conclusion, designers should not judge the key space of their proposed image encryption scheme based on the length of the external secret key solely.

CHAPTER 9

CRYPTANALYSIS OF A NEW IMAGE ALTERNATE ENCRYPTION ALGORITHM BASED ON CHAOTIC MAP

Typically, an image encryption scheme is said to have a good resistivity against differential attack based on two quantitative measures, i.e., the number of pixel change rate and the unified average changing intensity. However, these two quantitative measures are not a guarantee that the scheme is secure. In this chapter, we analyse the security of a new image alternate encryption scheme proposed by X. Wang and Guo, especially from cryptographic point of view, in line with the designers' approach in their security analyses. Negatively, even though that the experiment results showed that the aforementioned quantitative measures are close to the ideal values, the image encryption scheme is still vulnerable to an impossible differential attack on 9-round X. Wang and Guo scheme. This contradicts the designers' claim that their proposed image encryption scheme is secure against chosen plaintext attack even for r as small as 2. Lastly, a simple divide-and-conquer attack can be applied to their proposed scheme by obtaining the encrypted image of a large all black image.

9.1 Introduction

In 2003, Pareek, Patidar and Sud (2003) proposed a symmetric key block cipher algorithm based on discrete chaos cryptography. Since then, a number of proposals had been presented to encrypt plain images using chaos cryptography, this includes but is not limited to (Mirzaei, Yaghoobi, & Irani, 2012; Pareek et al., 2005, 2006; Pareek, Patidar, & Sud, 2009, 2013; Patidar, Pareek, Purohit, & Sud, 2010; Tong, Wang, Zhang, & Liu, 2013). However, many proposed image encryption schemes are not well studied against cryptanalytic attacks and subsequently had been found insecure against various attacks (Álvarez, Montoya, Romera, & Pastor, 2003; Li, Li, Álvarez, Chen, & Lo, 2008; Li et al., 2009; Li, Liu, Xie, & Chen, 2013; Li, Zhang, Ou, Wong,

& Shu, 2012; X. Wang & Liu, 2013; Zhang & Wang, 2014). Many proposals share a similarity where the designers claimed that their schemes are secure against differential attack based on two quantitative measures, i.e., the number of pixel change rate (NPCR) and the unified average changing intensity (UACI). If these two quantitative measures (which measure the influence of a 1-pixel change in a plain image P on the encrypted image C) are close to the ideal values, such scheme is said to have good property in resisting differential attacks.

Very recently, X. Wang and Guo (2014) proposed a new image alternate encryption scheme based on chaotic map. Technically, they proposed the use of key-dependent chaotic sequence to provide confusion and diffusion that are needed to result in a secure block cipher. X. Wang and Guo's proposed image encryption scheme can be treated as a block cipher since same round function $\mathbf{f}$ is iterated for r times to encrypt a plain image. X. Wang and Guo claimed that their proposed scheme is secure against chosen plaintext/plain image attack and is sensitive to the changing of secret key, even if the changes are very small. Besides, their experiment results showed that the measured NPCR and UACI are close to the ideal values when $r > 1$ and thus they claimed that their proposed scheme can resist plaintext and differential attack effectively when $r \geq 2$.

In this chapter, we revisit the security of X. Wang and Guo image encryption scheme using some systematic cryptanalytic approaches. Our contributions are listed as follows:

1. We study the key space of X. Wang and Guo image encryption scheme in detail.
2. We present a variant of differential attack, namely impossible differential attack, on r-round X. Wang and Guo image encryption scheme where $2 \leq r \leq 9$.
3. We present a divide-and-conquer attack on X. Wang and Guo image encryption scheme by exploiting a large all black image (i.e., $P(i) = 0$ for $i = 1, 2, \ldots, m \times n$).

These results violate the claims stated by X. Wang and Guo (2014).

Organisation. The remainder of this chapter is organised as follows. In the next section, we describes the specification of X. Wang and Guo image alternate encryption algorithm from a cryptographic perspective.We first analyse the key space implied by the chosen parameters in X. Wang and Guo image encryption scheme in Section 9.3. In Section 9.4, we present an impossible differential attack on r-round X. Wang and Guo image encryption algorithm based on miss-in-the-middle approach. Section 9.5 shows a divide-and-conquer attack on r-round X. Wang and Guo image encryption scheme by exploiting a large all black image. Section 9.6 concludes the chapter.

9.2 The X. Wang and Guo Image Encryption Scheme

The image encryption algorithm proposed by X. Wang and Guo is interpreted from a cryptographic perspective for ease of understanding. Given an $m \times n$ image P, which in vectorised form is denoted as $P(i)$, for $i = 1, 2, \ldots, m \times n$. P can be treated as an image which contains i pixels. The image encryption scheme outputs an encrypted image C on input an image P and a secret key $K = (u, u', y(0))$ where $y(0) \in (0,1)$ and $u, u' \in [0,4]$. The image encryption scheme can be divided into three main algorithms, i.e., `Key Schedule`, `E` and `D`. We ignore the details of `D` algorithm as it is the inverse of `E` algorithm.

9.2.1 The `Key Schedule` Algorithm

This algorithm on input part of a secret key $(u', y(0))$ computes the chaotic sequence $y = \{\{y(i)\}\}_{i=1}^{m \times \frac{n}{2}}$ using the following logistic map

$$y(i+1) = u' \cdot y(i) \cdot (1 - y(i)), y(i) \in (0,1), u' \in [0,4] \tag{9.1}$$

where $i = 0, 1, 2, \ldots, (m \times \lceil n/2 \rceil) - 1$. May (1974) proved that the logistic map is chaotic for $3.5699456 < u' \leq 4$. Thus, throughout this chapter, we assume $3.5699456 < u' \leq 4$ to result a chaotic map.

9.2.2 The E Algorithm

1. **Setup**:

   a) Compute the average $\mathsf{avg} = \dfrac{\mathsf{sum}(P)}{m \times n}$ where sum denotes the sum of all pixel values $P(i)$ in P; and obtain its decimal part $\mathsf{avg1} = \mathsf{avg} - \lfloor \mathsf{avg} \rfloor$.

   b) Compute $x(0) = \mathsf{avg}_2 = \dfrac{\lfloor \mathsf{avg}_1 \times 10^6 \rfloor}{10^6}$.

   c) Compute the sequence $z = \{z(i)\}_{i=1}^{m \times \frac{n}{2}}$ such that $z(i) = \lfloor \mod (y(i) \times 10^{10} + \mathsf{sum}(P) \times 10^3, 256) \rfloor$ where $i = 1, 2, \ldots, m \times \lceil n/2 \rceil$ and $\mod(\cdot, 256)$ denotes modulo reduction by the modulus 256.

   d) Compute $rk = \mathsf{reshape}(z, m, \lceil n/2 \rceil)$ where $\mathsf{reshape}(\cdot, i, j)$ denotes the reverse of vectorisation i.e., converting the vector back into matrix form of dimension $i \times j$.

2. **Encrypt**: Given an $m \times n$ plain image P, if n is an odd number, another known column should be concatenated to P such that the column n becomes even. P is then processed iteratively for r rounds (as shown in Figure 9.1), where each round i performs the following:

   a) P is column-wise divided into two equal-sized blocks, each of length $\lceil n/2 \rceil$, denoted respectively as left half block XL_i and right half block XR_i.

   b) Compute XR_{i+1} as the XOR of XL_i with rk_i, i.e. $XR_{i+1} = XL_i \oplus rk_i$. Note that $rk_i = rk$ and rk is from Step 1(d).

   c) Shuffle XR_{i+1} to obtain h as follows:

      i. Generate the chaotic sequence $x = \{x(i)\}_{i=1}^{m \times \frac{n}{2}}$ using the following logistic map

$$x(i+1) = u \cdot x(i) \cdot (1 - x(i)), x(i) \in (0,1), u \in [0,4] \tag{9.2}$$

      where $i = 0, 1, 2, \ldots, (m \times \lceil n/2 \rceil) - 1$ and $x(0)$ is from Step 1(b).

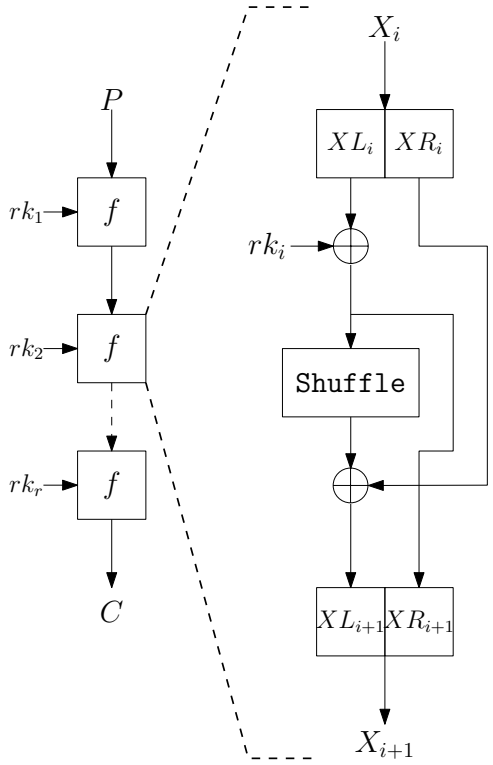


Figure 9.1: Graphical Presentation of Encrypt with the Number of Rounds r

ii. Change the sequence $x(i)$ into a $\{0,1\}$ sequence by thresholding, i.e., $x(i) = 1$ if $x(i) \geq 0.5$, and $x(i) = 0$ if $x(i) < 0.5$ for $i = 1, 2, \ldots, (m \times \lceil n/2 \rceil)$.

iii. Initialise three arrays M, N, R. For $i = 1, 2, \ldots, m \times \lceil n/2 \rceil$, based on the value of $x(i)$, copy $P(i)$ into M if $x(i) = 1$, else copy $P(i)$ into N. Concatenate M and N into R as follows. If the number of rounds r is odd, then $R = M||N$, else if r is even, then $R = N||M$ where $||$ denotes concatenation.

iv. Compute $h = \mathsf{reshape}(R, m, \lceil n/2 \rceil)$.

d) Compute XL_{i+1} as the XOR of XR_i with h, i.e. $XL_{i+1} = XR_i \oplus h$.

e) Concatenate XL_{i+1} and XR_{i+1} to form $X_{i+1} = XL_{i+1}||XR_{i+1}$.

As mentioned by X. Wang and Guo (2014), $x(0) = \mathsf{avg}_2$ and thus it depends on $\mathsf{sum}(P)$, which in turn depends on the plaintext P. In the chosen plaintext attack

scenario (Goldwasser & Micali, 2014) considered by X. Wang and Guo, the attacker is given the ability to request the encrypted image C for any chosen plain image P of the attacker's choice. Thus, $\mathsf{sum}(P)$ and $x(0)$ would be known since the image P is chosen by the attacker and thus shall not be considered as part of a secret key. Throughout this chapter, we may use chosen plain image and encrypted image pairs to denote chosen plaintext pairs.

9.3 Revisiting the Key Space of X. Wang and Guo Image Encryption Scheme

According to IEEE 754-1985 standard (2008), the computational precision of the 64-bit double precision floating point number is approximately with 15 decimal digits. X. Wang and Guo presented a rough key space analysis. More precisely, they claimed that the key space was almost 10^{48} given that $u, u', y(0)$ and a single digit positive integer r can be guessed roughly with the probability of $10^{-16}, 10^{-16}, 10^{-15}$ and 10^{-1} respectively.

Since T is a single digit (i.e., 0 to 9) and r must be greater than 2 to resist plaintext and differential attacks, thus we have that $2 \leq r \leq 9$ which contains only 3 bits of entropy. Besides, $u, u' \in (3.5699456, 4]$ to result a chaotic logistic map and both u and u' are with 15 decimal places, thus u (or u') can be guessed with the probability of $\frac{1}{(4-3.5699456)\times 10^{15}} \approx 2^{-48.612}$. Similarly, $y(0) \in (0,1)$ is with 15 decimal places and can be guessed with the probability of $\frac{1}{10^{15}} \approx 2^{-49.829}$. Finally, the secret key $K = \langle u, u', y(0), T \rangle$ can be guessed with the probability of $2^{2\times(-48.612)+(-49.829)+(-3)}$ = $2^{-150.053}$ as compared to $10^{-48} \approx 2^{-159.453}$ shown by X. Wang and Guo (2014). After the key space is analysed in detail, the brute-force attack on X. Wang and Guo image encryption scheme has a time complexity of $2^{150.053}$ encryptions as compared to $2^{159.453}$ encryptions claimed by X. Wang and Guo. Thus, an attacker aims to break an image encryption algorithm with the time complexity less than the brute-force attack. The existence of such attacks with time complexity better than brute-force attack will indicate that such image encryption algorithm is insecure.

9.4 The Impossible Differential Attack on X. Wang and Guo Image Encryption Scheme

In this section, we first present an i-round differential with probability of one based on the number of pixel's differences. Subsequently, we show that such i-round differential can be used to construct an impossible differential based on miss-in-the-middle approach (Biham et al., 1999b). Finally, such impossible differential can be exploited to launch an impossible differential attack on X. Wang and Guo image encryption scheme for $2 \leq r \leq 9$.

9.4.1 The i-Round Differential with F

We present an i-round differential of the X. Wang and Guo proposed image encryption scheme with probability of one (see Figure 9.2 for the graphical presentation of i-round differential). Given two plain images P and P', the input difference equals $P \oplus P'$ and the output difference equals $C \oplus C'$ where $\oplus$ denotes bitwise exclusive-or operation. Note that C and C' are obtained by encrypting P and P' respectively using a single unknown secret key K.

Let Hamming weight of a difference of α between two input images X and X', $HW(\alpha) = j$ such that $\#\{(X(i) \oplus X'(i)) \neq 0\} = j$ for $i = 1, \ldots, m \times n/2$ and $j \geq 0$ where $X(i)$ and $X'(i)$ represent the i-th pixel of the image X and X' respectively.

Since single unknown secret key $K = \langle u, u', y(0), r \rangle$ is used to encrypt two different plain images P and P' such that $\mathsf{sum}(P) = \mathsf{sum}(P')$, $rk'_i = rk_i$ for all Round i. Thus, in each round, the same value of rk will be XORed with the left half of data. Besides, the shuffling step is a permutation that depends on the value of $x(0)$ (which is based on $\mathsf{sum}(\cdot)$). Since $\mathsf{sum}(P) = \mathsf{sum}(P')$, the shuffling step will be the same permutation for both images P and P'. We exploit these observations to construct the following i-round differential of the X. Wang and Guo image encryption scheme with probability of one.

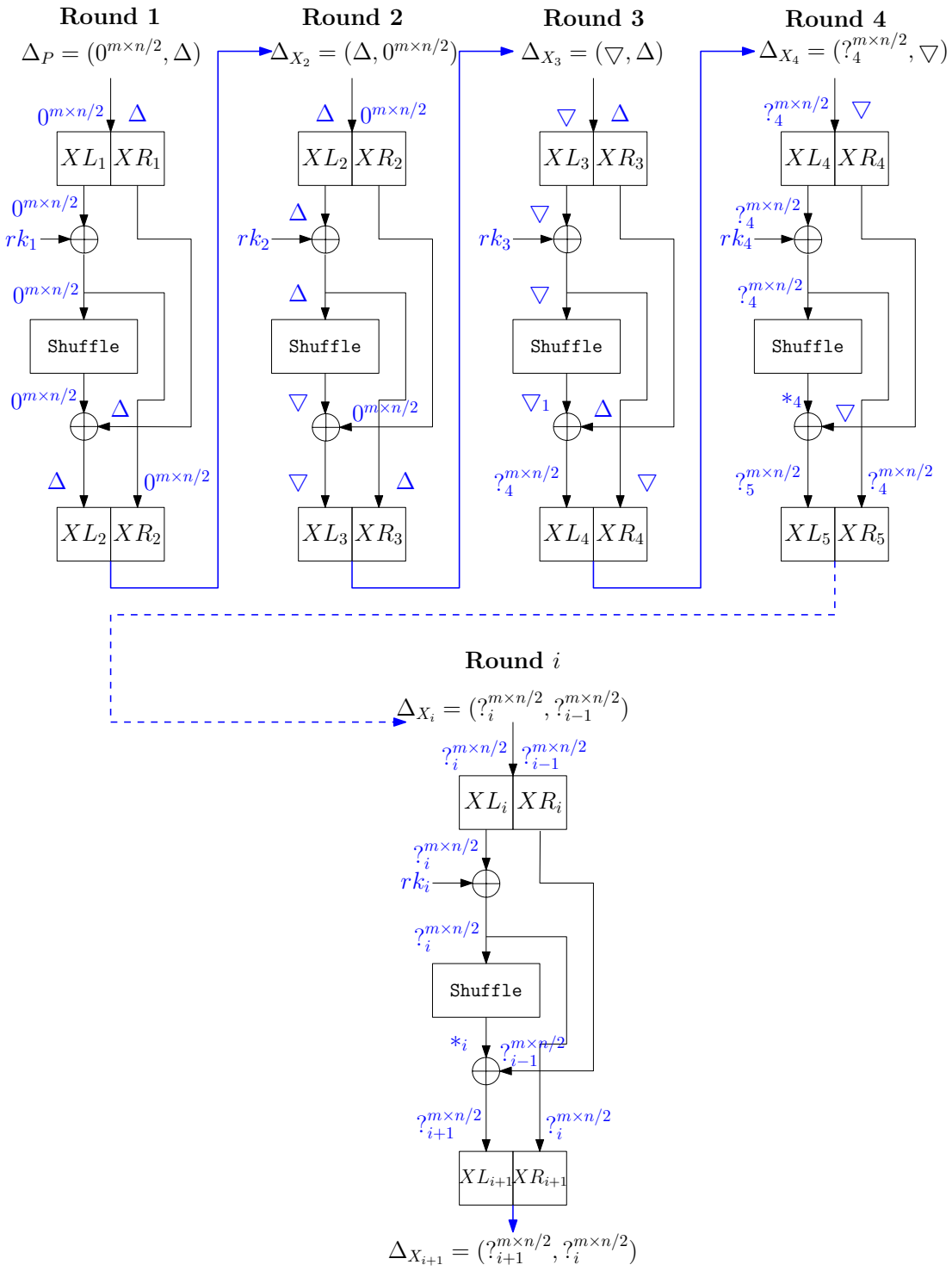


Figure 9.2: The i-Round Differential of the X. Wang and Guo Image Encryption Scheme with Probability of One

1. Choose two plain images $P = (XL_1, XR_1)$ and $P' = (XL'_1, XR'_1) = (XL_1, XR_1 \oplus \triangle)$ keyed under the same unknown key, such that $\mathsf{sum}(P) = \mathsf{sum}(P')$. For instance, two images can have right halves that differ and yet have their different pixel values sum to the same amount, therefore $rk'_1 = rk_1$, and we have $XR'_2 = XR_2$ and $h' = h$.

2. We then have $XL'_2 \oplus XL_2 = XR'_1 \oplus XR_1 = XR_1 \oplus \triangle \oplus XR_1 = \triangle$, and therefore at the end of Round 1, we have the output difference $(\triangle, 0^{m \times n/2})$.

3. Going into the second round, we have that a difference $\triangle$ goes into the shuffling step, producing some difference $\triangledown$.

4. Subsequently, the output difference after Round 2 is $(\triangledown, \triangle)$.

5. Due to the observation that $HW(\triangle) = HW(\triangledown)$ as the shuffling step is a $\mathsf{sum}(\cdot)$-dependent permutation and $\mathsf{sum}(P) = \mathsf{sum}(P')$, we extend the previous stated two-round differential to four-round differential with probability of 1.

6. Some important observations from the four-round differential are $HW(?_4) \leq 2 \times HW(\triangle)$ and $HW(?_5) \leq 3 \times HW(\triangle)$ where the question marker $?_j$ represents an indeterminate difference for any integer j. This is because $\triangledown$ may equal $\triangle$.

7. Continuing the same strategy, we can construct any-round differential with probability of 1. The output difference after Round i is $(?_{i+1}, ?_i)$ where $HW(?_{i+1}) \leq (i-1) \times HW(\triangle)$ and $HW(?_i) \leq (i-2) \times HW(\triangle)$ even though the specific differences of $?_{i+1}$ and $?_i$ remain unknown.

The limitation of this type of differential is $(i-2) \times HW(\triangle) < (m \times n/2)$, else one can no longer distinguish this pattern. For example, let $m = 6$, $n = 4$ and $HW(\triangle)_{min} = 2$ such that $\mathsf{sum}(P) = \mathsf{sum}(P')$, $i_{max} = 7$ where i_{max} refers to the maximum number of rounds such that the differential is still with probability of 1. To simplify the description, the above i-round differential is denoted as $\langle 0, \triangle \rangle \xrightarrow{i\text{-r}} \langle ?_{i+1}, ?_i \rangle$.

9.4.2 The Impossible Differential Attack

To initiate an impossible differential attack on r-round X. Wang and Guo image alternate encryption scheme, a $(r-1)$-round impossible differential can be exploited and be constructed using the miss-in-the-middle approach (Biham et al., 1999b).

We give an concrete example in constructing an impossible differential attack on 9-round X. Wang and Guo image encryption scheme since $r_{max} = 9$, however the same attack approach can be generalised to attack 2- to 8-round X. Wang and Guo image encryption scheme.

For $T = 9$, a straightforward approach in constructing an 8-round impossible differential is to seek for i-round and j-round differentials with probability of one where $i + j = 8$. The other compulsory condition is that the output difference of i-round differential must contradict with the input difference of j-round differential, i.e., the probability that i-round differential leads to j-round differential is 0. Hence, such differential is named as an impossible differential.

We use a 7-round differential with probability of one, i.e., $\langle 0, \triangle \rangle \xrightarrow{\text{7-r}} \langle ?_8, ?_7 \rangle$ and an 1-round differential with probability of one, i.e., $\langle 0, \triangle^* \rangle \rightarrow \langle \triangle^*, 0 \rangle$ as the building blocks in constructing our 8-round impossible differential to attack 9-round image alternate encryption algorithm. Besides, let $HW(\triangle) = 2$ by choosing appropriate images P and P' such that $\mathsf{sum}(P) = \mathsf{sum}(P')$. Note that $HW(?_8) \leq 6 \times HW(\triangle) \leq 12$ and $HW(?_7) \leq 5 \times HW(\triangle) \leq 10$. If $HW(\triangle^*) > 10$, one may notice that the output difference of 7-round differential with probability of one contradicts with the input difference of 1-round differential with probability of one, i.e., $HW(?_7) \neq HW(\triangle^*)$. Thus, an 8-round impossible differential ($\langle 0, \triangle \rangle \xrightarrow{\text{8-r}} \langle \triangle^*, 0 \rangle$ with probability of zero) is formed and can be exploited to recover the unknown secret key $K = \langle u, u', y(0), r \rangle$ using the following attack procedure.

We give an example by setting $m = 6$ and $n = 4$. Note that to obtain the input difference of $\langle \triangle^*, 0 \rangle$ in Round 9, the output difference in Round 9 should equal

$\langle \bigtriangledown^*, \bigtriangleup^* \rangle$ where $HW(\bigtriangleup^*) = HW(\bigtriangledown^*) > 10$.

1. Select 2^ϕ structures $\mathscr{S}_i$ (a specific value of ϕ will be given below and for $i = 1, 2, \cdots, 2^\phi$) where a structure is defined to be a set of 2^{16} chosen plain images such that all the plain images P have the same value in all pixels excepts 2 pixels in the right half with fixed positions. In a chosen plaintext attack scenario, obtain all the encrypted images for the 2^{16} plain images in each of the 2^ϕ structures.

2. For each structure, perform the following sub-steps:

   a) Fix $\mathsf{sum}(P) = c + 255$ where c is the sum of all the pixels except the 2 pixels in the right half with fixed positions. From the computer simulation, there are $256 = 2^8$ out of 2^{16} plain images satisfying this condition. The expected remaining chosen plaintext-ciphertext pairs in a structure is 2^8. We denote $C_{i,j}$ as the encrypted image for plain image $P_{i,j}$ for $j = 1, 2, \ldots, 2^8$.

   b) Given the remaining 2^8 chosen plaintext-ciphertext pairs in a structure, for $1 \leq l \leq 2^8$, generate all the $\frac{2^8 \times (2^8 - 1)}{2} \approx 2^{15}$ possible chosen plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ where$P_{i,j} \neq \widehat{P}_{i,l}$, $\mathsf{sum}(P_{i,j}) = \mathsf{sum}(\widehat{P}_{i,l})$ and $HW(P_{i,j} \oplus \widehat{P}_{i,l}) = 2$.

   c) Given the remaining 2^{15} chosen plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ in a structure, select those quartets such that $C_{i,j} \oplus \widehat{C}_{i,l}$ have the form of $(\bigtriangledown^*, \bigtriangleup^*)$ where $HW(\bigtriangledown^*) = HW(\bigtriangleup^*) \geq 11$. Note that both halves have $6 \times 2 = 12$ pixels. The expected remaining chosen plaintext-ciphertext quartets in a structure is $2^{15} \times \sum_{s=11}^{12} \binom{12}{s}^2 / 2^{2 \times 12} = 2^{15} \times 145 / 2^{24} = 2^{-1.82}$.

   d) Compute $x(0)$. Note that $x(0)$ is same for any remaining plain images since sum is fixed.

   e) Guess a value for $2^{48.612}$ possible key u and perform the following sub-steps:

      i. Given the remaining plaintext-ciphertext quartets $(P_{i,j}, C_{i,j}, \widehat{P}_{i,l}, \widehat{C}_{i,l})$ in a structure, shuffle right half of $C_{i,j}$ and $\widehat{C}_{i,l}$ to obtain h and $\widehat{h}$ respectively.

ii. Check whether $h \oplus \widehat{h} = \bigtriangledown^*$. If equal, then the guess of u is wrong and thus can be filtered out.

Analysis. The attack requires $2^{\phi} \times 2^{16}$ chosen plain images. Step 1 has a time complexity of $2^{\phi+16}$ 9-round encryptions. For a structure of 2^{16} chosen plain images, Step 1 has a memory complexity of $2^{16} \times (2)$ bytes $\approx 2^{17}$ bytes since all other pixels are fixed to a certain value. The memory complexity of Step 2(b) is $2^{15} \times (2+24)$ bytes $\approx 2^{19.7}$ bytes to store all possible chosen plaintext-ciphertext quartets. Besides, Step 2(b) has a time complexity of about 2^{15} memory accesses to generate 2^{15} chosen plaintext-ciphertext quartets.

Next, Step 2(c) has a time complexity of about 2^{15} memory accesses to access 2^{15} chosen plaintext-ciphertext quartets. Lastly, for a structure of $2^{-1.82}$ chosen plaintext-ciphertext quartets, Step 2(e) has a time complexity of about $2^{-1.82} \times 2^{48.612} \times (2$ Shuffle + 1 XOR) operations. Since one-round encryption requires 1 Shuffle and 2 XOR operations, thus we consider the sum of 2 Shuffle and 1 XOR operations equals the time complexity of two-round encryption. Step 2(e) has a time compexity of $2^{-1.82} \times 2^{48.612} \times \frac{2}{9} \approx 2^{44.622}$ 9-round encryptions.

In total, the impossible attack has a time complexity of approximately $2^{\phi+16} + 2^{\phi} \times (2^{44.622}) \approx 2^{\phi+44.622}$ 9-round encryptions and $2^{\phi} \times (2^{15} + 2^{15}) = 2^{\phi+16}$ memory accesses. An extremely conservative estimate is that 9 memory accesses equal a 9-round encryption in terms of time assuming that the E function with a round subkey is precomputed in a table and is equivalent to one memory access by neglecting the computational complexity for the key schedule. Thus, one round is equivalent to one memory access. As a consequence, the total time complexity of the impossible differential attack is $2^{\phi+44.622} + \frac{2^{\phi+16}}{9} \approx 2^{\phi+44.622}$ 9-round encryptions.

The attack succeeds if the expected number of u that will not be filtered out after performing the above attack procedure is less than 1 as one out of the $2^{48.612}$ possible values of u must be correct. In the beginning of Step 2(e), the expected

remaining chosen plaintext-ciphertext quartets in all the structures is $2^{\phi-1.82}$. There exists $2^{48.612}$ permutations due to $2^{48.612}$ possible values of u. Since an image has a $6\times 4 = 24$ pixels, each half consists of 12 pixels. Thus, the Shuffle operation is a 12×12-pixel permutation and there exists $12! = 2^{28.835}$ different 12×12-pixel permutations. However, due to $2^{48.612}$ possible values of u, then $2^{19.777}$ different values of u will generate the same permutation assuming that the permutation generated based on u is uniformly distributed.

Observe that given any guess of u, the probability that $h \oplus h' = \bigtriangledown^*$ is $2^{-28.835}$ (one out of $2^{28.835}$ permutations) and thus the probability that each of the $2^{28.835}$ possible values of u is filtered out using one of $2^{\phi-1.82}$ possible chosen plaintext-ciphertext quartets is $2^{-28.835}$. It follows that the expected number of u that will not be filtered out after performing the above attack procedure is $2^{28.835} \times (1-2^{-28.835})^{2^{\phi-1.82}}$. Thus, by letting $\phi = 50.655$, only $2^{19.777}$ of the $2^{48.612}$ possible values of u will remain.

As a conclusion, the impossible differential attack has a time complexity of $2^{\phi+44.622} = 2^{95.277}$ 9-round encryptions, a memory complexity of $2^{51.215}$ bytes (which is dominated by the step in storing the list of filtered wrong key bytes) and a data complexity of $2^{\phi+16} = 2^{66.655}$ chosen plaintexts. Equipped with $2^{19.777}$ possible values of u, the attacker can recover the remaining secret key $\langle u', y(0), T\rangle$ using brute-force attack with time complexity of $2^{121.218}$ 9-round encryptions.

Remark: The above attack procedure can be generalised to attack X. Wang and Guo image encryption scheme for $T = 2$ to 8 as the attacker can always construct a j-round impossible differential (i.e., $\langle 0, \triangle\rangle \xrightarrow{j\text{-r}} \langle \triangle^*, 0\rangle$ with probability of 0) for $j = 1$ to 8 by letting $HW(\triangle) = 2$ and $HW(\triangle^*) > 10$. By using different size of images, one may obtain better attack in terms of time complexity. We leave the details to interested readers.

9.5 A Divide-and-Conquer Attack using A Large All Black Image

When an all black image P (i.e., $P(i) = 0$ for $i = 1, 2, \ldots, m \times n$) is encrypted, $\mathsf{sum}(P) = 0$ and thus $x(0) = 0$ for $3.5699456 < u_i \leq 4$ for $i \geq 0$. This contradicts with Guo and Wang's description that $x(0) \neq 0$. In this scenario, an all black image P will be encrypted to the same encrypted image C if all the secret keys have the same value of $u', y(0)$ and r regardless any value of u.

We exploit the above observation to launch a divide-and-conquer attack on X. Wang and Guo image encryption scheme. The attacker may use the following attack procedure to find the secret key $K = \langle u, u', y(0), r \rangle$ with time complexity lesser than a brute-force attack.

Let C be the encrypted image of an all black image P under the secret key k and each pixel is of 1 byte. Besides, the attacker is assumed to have plaintext pairs of (P_i, C_i) for $i \geq 1$ and C_i is the encrypted image of P_i under the secret key K. Note that $P_i \neq P$ and each P_i is different for different value of i.

1. Randomly fix a value for u as it will not affect the $\mathtt{E}$ function. This is because for an all black image P, $\mathsf{sum}(P) = 0$ and thus $x(0) = 0$ for any value of u. Similarly, $x(i) = 0$ for $i \geq 1$ which indicates the same permutation will be applied for any value of u.
2. Guess a value for 8 possible values of r and perform the following sub-steps:
   a) Guess a value for $2^{48.612} \times 2^{49.829} = 2^{98.441}$ of u' and $y(0)$, perform the following sub-steps:
      i. Encrypt P to generate the encrypted image C' using the guessed value of $u', y(0)$ and T.
      ii. If $C = C'$, store the candidate of $\langle u', y(0) \rangle$.
   b) For the remaining $2^{98.411}/(m \times n)$ bytes $= 2^{95.411-|m|-|n|}$ possible values of $\langle u', y(0) \rangle$ where $|i|$ denotes the length of i in bits, guess a value for $2^{48.612}$

possible values of u and perform the following sub-steps:

i. Encrypt P_1 to generate the encrypted image C_1' using the guessed value of $u, u', y(0)$ and r.

ii. Discard the secret key $k' = \langle u, u', y(0) \rangle$ if $C_1' \neq C_1$.

iii. If $C_1' = C_1$, the total remaining possible values of secret key is $2^{92.411-2|m|-2|n|}$. Repeat the Step 2(b)(ii) and Step 2(b)(iii) until one unique value of secret key remains using different plaintext P_i.

c) Use one additional plaintext-ciphertext pair (P'', C'') and check whether the remaining secret key is the right key. If not, then the guessed r is wrong and repeat the above steps with other value of r.

Analysis. As a concrete example, if a plain image P with high resolution has 16384×16384 pixels (i.e., such image is with $2^{14} \times 2^{14} \times 2^3$ bits = 256 MB), then Step 2(a) has the time complexity of $2^{98.411}$ encryptions and needs one pair of (P, C). The total possible values of secret keys that passes Step 2(a) is $2^{98.411-3-14-14} = 2^{67.411}$. In Step 2(b), there exist $2^{67.411+48.612} = 2^{116.023}$ possible values of secret keys. Thus, Step 2(b) needs 4 additional plain image and encrypted image pairs (i.e., $(P_1, C_1), \ldots, (P_4, C_4)$) to reduce $2^{116.023}$ possible values of secret keys to only one unique secret key. The Step 2(b) has the time complexity of $2^{116.023} + 2^{85.023} + 2^{54.023} + 2^{23.023} \approx 2^{116.023}$ encryptions. Lastly, consider the worst case where the attack procedure is repeated for 8 times (i.e., r = 2 to 9), the overall time complexity is around $8 \times 2^{116.023} = 2^{119.023}$ encryptions. Meanwhile, the total plaintext-ciphetext pairs needed is 6. Note that the efficiency of the attack greatly depends on the size of underlying plain image. Due to the emerging of technology, the size of an image becomes larger from time to time (e.g., medical imaging and pattern recognition) and thus the attack will become faster and needs lesser plain image and encrypted image pairs.

9.6 Summary

Both presented impossible differential and divide-and-conquer attacks on r-round X. Wang and Guo image encryption scheme for $2 \leq r \leq 9$ justify that the se-

curity of X. Wang and Guo image encryption scheme had been over-estimated by the designers. Moreover, impossible differential attack is a kind of chosen plaintext attack being examined and considered by X. Wang and Guo (2014). Many image encryption schemes are said to have good property against differential attack by showing the measured NPCR and UACI are close to the ideal values. However, our result shows that these two values are not adequate to conclude that an image encryption scheme is secure against impossible differential attack. We note that even the showed attacks are still of high complexity, but these attacks are faster than brute-force attack up to around $2^{28.835} \approx 478$ million times. Through our observations and attacks, we conclude that it may not be a good idea to design a round function that depends on two parts of secret key, e.g., $\langle u', y(0) \rangle$ and $\langle u \rangle$ while these two parts of secret key affect the round function independently. Lastly, the designer shall not justify that the proposed image encryption scheme has good property to resist differential attack based on two quantitative measures (i.e., NPCR and UACI).

CHAPTER 10

CONCLUSION AND FUTURE WORK

In this chapter, we summarise the main contributions and results of this thesis. We then wrap up this thesis with some suggestions for future research work.

10.1 Conclusion

The cryptanalytic results obtained in this thesis on block ciphers and image encryption schemes are listed as follows.

- We have presented related-key differential and related-key amplified boomerang attacks on the full MISTY1 under certain weak key assumptions.
- We have exhibited two 7-round differentials with probabilities strictly bigger than the best previously known one on the SEED cipher and presented a differential cryptanalysis attack on a 9-round reduced version of SEED.
- We have demonstrated an impossible differential attack on the full CHAIN cipher for variable block length based on a generalised impossible differential.
- We have shown the weaknesses of the `Key Schedule` algorithms proposed in two recent image encryption schemes (Norouzi & Mirzakuchaki, 2014; X. Wang & Wang, 2014) where the effective space spanned by the subkeys generated from a secret key is a much smaller subset as compared to the secret key space.
- We have presented the first impossible differential attack on an image encryption scheme (X. Wang & Guo, 2014) which subsequently disproved a general assumption that an image encryption scheme is said to possess good properties to resist differential attack based on two quantitative measures. This finding also

suggests that the rules to design an image encryption scheme must be assessed from cryptographic perspectives as well.

The results exhibit the first concrete cryptographic weaknesses in the full MISTY1 cipher, full CHAIN cipher, X. Wang and Wang image encryption scheme, Norouzi and Mirzakuchaki image encryption scheme and Wang and Guo image encryption scheme. In addition, our result on SEED is better than any previously published cryptanalytic results on this cipher in terms of the number of attacked rounds and it suggests for the first time that the safety margin of SEED decreases below half of the number of rounds. Such results are important as both SEED and MISTY1 are ISO standards.

In the context of message authentication codes and modes of operation, we have analysed the security of GMAC and GCM with respect to the forgery and distinguishing attacks. More precisely, we obtained the following results on GMAC and GCM:

- We have generalised the set of weak key classes proposed by Saarinen (2012) to include all subsets of nonzero keys. Hence, we have removed the condition on the smoothness of $2^n - 1$, where n denotes the block size, for the existence of weak key classes.
- By considering powers of suitable field elements and linearised polynomials, we have exploited some specific weak key classes to present a universal forgery attack on GMAC.
- By invoking the birthday paradox arguments, we have showed that a chosen message attack can be used to distinguish GMAC from a random function.
- To relax the assumptions required in the universal forgery attack, we have demonstrated that we can utilise the uniqueness of the counter mode encryption to launch a known ciphertext attack against GCM itself when the initial vector (IV) is restricted to 96 bits.

We remark that the first three attack techniques can be applied to all Wegmen-Carter polynomial message authentication codes. Parts of the results of this thesis were independently found by Procter and Cid (2013) and McGrew (2012). McGrew named the uniqueness of the counter mode that can be exploited to launch a known ciphertext attack on the counter mode encryption as impossible plaintext cryptanalysis. Note that GCM is an ISO standard and widely used in real-life security protocols.

In addition, we have presented weaknesses of the PMAC schemes arising from two main characteristics of the constants in the construction. These weaknesses allow us to launch forgery attacks on different PMAC versions. However, we emphasise that our attacks did not break the security bound of PMACs. Rather, we provided explicit attacks which achieve these bounds in some instances. Note that PMAC is part of OCB which is an ISO standard.

10.2 Future Work

In view of the research conducted in this thesis, we suggest some possible interesting directions to cryptanalyse block ciphers and block cipher based constructions:

- The results we obtained against MISTY1 rely on weak key and related-key assumptions and with high complexity. It would be better if a new attack could be mounted with a practical time complexity under a single unknown secret key scenario.

- It would be interesting to explore whether some good cryptanalytic results could be obtained when we apply the attack techniques on GCM to other Wegmen-Carter polynomial message authentication codes and some real-world applications that make use of the counter mode encryption. Similarly, we may explore whether the observations we obtained on PMAC could be applied to the applications that make use of the same constant generation method, especially to the authenticated encryption candidates submitted to the CAESAR competition (e.g., AES-OTR (Minematsu, 2015), AES-COPA (Andreeva et al., 2015) and

SHELL (L. Wang, 2015)).

- Image encryption is a new but fast moving field. From the results obtained in this thesis, we may conclude that the security of an image encryption scheme is not analysed from widely studied cryptanalytic perspectives. Applying cryptanalytic attacks on image encryption schemes may lead to new design rules for an image encryption scheme. In addition, the analysis of the subkey space can be extended to any image encryption scheme even though it does not require an external secret key.

REFERENCES

[1] 3GPP. (2001). Technical specification group services and system aspects, 3G security, specification of the 3gpp confidentiality and integrity algorithms; document 2: KASUMI specification, v3.1.1.

[2] Adams, C. M. (1997). Constructing symmetric ciphers using the cast design procedure. *Designs, Codes and Cryptography*.

[3] Álvarez, G., Montoya, F., Romera, M., & Pastor, G. (2003). Cryptanalysis of a discrete chaotic cryptosystem using external key. *Physics Letters A*, *319*(3-4), 334-339. doi: 10.1016/j.physleta .2003.10.044

[4] Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., & Yasuda, K. (2015). AES-COPA v2. Retrieved from `https://competitions.cr.yp.to/round2/aescopav2.pdf` (https://competitions.cr.yp.to/round2/aescopav2.pdf)

[5] Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., & Tokita, T. (2001). Camellia: A 128-bit block cipher suitable for multiple platforms — design and analysis. In D. R. Stinson & S. Tavares (Eds.), *Proceedings of SAC '00 — The 7th Annual International Workshop on Selected Areas in Cryptography, Volume 2012 of Lecture Notes in Computer Science* (p. 39-56). Waterloo, Ontario, Canada: Springer Berlin Heidelberg. doi: 10.1007/3-540-44983-3 _4

[6] Babbage, S., & Frisch, L. (2000). On MISTY1 higher order differential cryptanalysis. In D. Won (Ed.), *Proceedings of ICISC '00 — The Third International Conference on Information Security and Cryptology, Volume 2015 of Lecture Notes in Computer Science* (p. 22-36). Seoul, Korea: Springer Berlin Heidelberg. doi: 10.1007/3-540-45247-8_3

[7] Bellare, M., Canetti, R., & Krawczyk, H. (1996). Keying hash functions for message authentication. In N. Koblitz (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '96 — The 16th Annual International Cryptology Conference, Volume 1109 of Lecture Notes in Computer Science* (p. 1-15). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-68697-5_1\clearpage

[8] Bellare, M., & Namprempre, C. (2000). Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto (Ed.), *Advances in Cryptology - Proceedings of ASIACRYPT '00 — The 6th International Conference on the Theory and Application of Cryptology and Information Security, Volume 1976 of Lecture Notes in Computer Science* (p. 531-545). Kyoto, Japan: Springer Berlin Heidelberg. doi: 10.1007/3-540-44448-3_41

[9] Bellare, M., & Namprempre, C. (2008). Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology*, *21*(4), 469-491. doi: 10.1007/s00145-008-9026-x

[10] Bernstein, D. J. (2005a). The Poly1305-AES message-authentication code. In H. Gilbert & H. Handschuh (Eds.), *Proceedings of FSE '05 — The 12th International Workshop on Fast Software Encryption, Volume 3557 of Lecture Notes in Computer Science* (p. 32-49). Paris, France: Springer Berlin Heidelberg. doi: 10.1007/11502760_3

[11] Bernstein, D. J. (2005b). Stronger security bounds for Wegman-Carter-Shoup authenticators. In R. Cramer (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '05 — The 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Volume 3494 of Lecture Notes in Computer Science* (p. 164-180). Aarhus, Denmark: Springer Berlin Heidelberg. doi: 10.1007/11426639_10

[12] Bierbrauer, J., Johansson, T., Kabatianskii, G., & Smeets, B. (1994). On families of hash functions via geometric codes and concatenation. In D. R. Stinson (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '93 — The 13th Annual International Cryptology Conference, Volume 773 of Lecture Notes in Computer Science* (p. 331-342). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-48329-2_4

[13] Biham, E. (1994). New types of cryptanalytic attacks using related keys. In T. Helleseth (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '93 — Workshop on the Theory and Application of of Cryptographic Techniques, Volume 765 of Lecture Notes in Computer Science* (p. 398-409). Lofthus, Norway: Springer Berlin Heidelberg. doi: 10.1007/3-540-48285-7_34

[14] Biham, E., Biryukov, A., & Shamir, A. (1999a). Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In J. Stern (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '99 — The International Conference on the Theory and Application of Cryptographic Techniques, Volume 1592 of Lecture Notes in Computer Science* (p. 12-23). Prague, Czech Republic: Springer Berlin Heidelberg. doi: 10.1007/3-540-48910-X_2

[15] Biham, E., Biryukov, A., & Shamir, A. (1999b). Miss in the middle attacks on IDEA and Khufu. In L. R. Knudsen (Ed.), *Proceedings of FSE '99 — The 6th International Workshop on Fast Software Encryption, Volume 1636 of Lecture Notes in Computer Science* (p. 124-138). Rome, Italy: Springer Berlin Heidelberg. doi: 10.1007/3-540-48519-8_10

[16] Biham, E., Dunkelman, O., & Keller, N. (2001). The rectangle attack — rectangling the Serpent. In B. Pfitzmann (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '01 — The International Conference on the Theory and Application of Cryptographic Techniques, Volume 2045 of Lecture Notes in Computer Science* (p. 340-357). Innsbruck, Austria: Springer Berlin Heidelberg. doi: 10.1007/3-540-44987-6_21

[17] Biham, E., Dunkelman, O., & Keller, N. (2005a). Related-key boomerang and rectangle attacks. In R. Cramer (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '05 — The 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Volume 3494 of Lecture Notes in Computer Science* (p. 507-525). Aarhus, Denmark: Springer Berlin Heidelberg. doi: 10.1007/11426639_30

[18] Biham, E., Dunkelman, O., & Keller, N. (2005b). A related-key rectangle attack on the full KASUMI. In B. Roy (Ed.), *Advances in Cryptology - Proceedings of ASIACRYPT '05 — The 11th International Conference on the Theory and Application of Cryptology and Information Security, Volume 3788 of Lecture Notes in Computer Science* (p. 443-461). Chennai, India: Springer Berlin Heidelberg. doi: 10.1007/11593447_24

[19] Biham, E., & Shamir, A. (1991a). Differential cryptanalysis of DES-like cryptosystems. In A. J. Menezes & S. A. Vanstone (Eds.), *Advances in Cryptology - Proceedings of CRYPTO '90 — The 10th Annual International Cryptology Conference, Volume 537 of Lecture Notes in Computer Science* (p. 2-21). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-38424-3_1

[20] Biham, E., & Shamir, A. (1991b). Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, *4*(1), 3-72. doi: 10.1007/BF00630563

[21] Biham, E., & Shamir, A. (1991c). Differential cryptanalysis of Feal and N-Hash. In D. W. Davies (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '91 — The Annual International Workshop on the Theory and Application of of Cryptographic Techniques, Volume 547 of Lecture Notes in Computer Science* (p. 1-16). Brighton, UK: Springer Berlin Heidelberg. doi: 10.1007/3-540-46416-6_1

[22] Biham, E., & Shamir, A. (1993). Differential cryptanalysis of the full 16-round DES. In E. F. Brickell (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '92 — The 12th Annual International Cryptology Conference, Volume 740 of Lecture Notes in Computer Science* (p. 487-496). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-48071-4_34

[23] Biryukov, A. (1999). Methods of cryptanalysis. *PhD Thesis, Technion – Israel Institute of Technology, Israel*.

[24] Biryukov, A., & Khovratovich, D. (2009). Related-key cryptanalysis of the full AES-192 and AES-256. In M. Matsui (Ed.), *Advances in Cryptology - Proceedings of ASIACRYPT '09 — The 15th International Conference on the Theory and Application of Cryptology and Information Security, Volume 5912 of Lecture Notes in Computer Science* (p. 1-18). Tokyo, Japan: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-10366-7_1

[25] Biryukov, A., Khovratovich, D., & Nikolić, I. (2009). Distinguisher and related-key attack on the full AES-256. In S. Halevi (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '09 — The 29th Annual International Cryptology Conference, Volume 5677 of Lecture Notes in Computer Science* (p. 231-249). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-03356-8_14

[26] Black, J. (2005). Authenticated encryption. *Encyclopedia of Cryptography and Security*.

[27] Black, J., Halevi, S., Krawczyk, H., Krovetz, T., & Rogaway, P. (1996). UMAC: Fast and secure message authentication. In M. Wiener (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '99 — The 19th Annual International Cryptology Conference, Volume 1666 of Lecture Notes in Computer Science* (p. 216-233). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-48405-1_14

[28] Black, J., & Rogaway, P. (2000). CBC MACs for arbitrary-length messages: The three-key constructions. In M. Bellare (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '00 — The 20th Annual International Cryptology Conference, Volume 1880 of Lecture Notes in Computer Science* (p. 197-215). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-44598-6_12

[29] Black, J., & Rogaway, P. (2001). A block-cipher mode of operation for parallelizable message authentication. *Cryptology ePrint Archive*. Retrieved from `http://eprint.iacr.org/2001/27` (http://eprint.iacr.org/2001/27)

[30] Black, J., & Rogaway, P. (2002). A block-cipher mode of operation for parallelizable message authentication. In L. R. Knudsen (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '02 — The International Conference on the Theory and Application of Cryptographic Techniques, Volume 2332 of Lecture Notes in Computer Science* (p. 384-397). Amsterdam, The Netherlands: Springer Berlin Heidelberg. doi: 10.1007/3-540-46035-7_25

[31] Blondeau, C., & Gérard, B. (2011). Multiple differential cryptanalysis: Theory and practice. In A. Joux (Ed.), *Proceedings of FSE '01 — The 18th International Workshop on Fast Software Encryption, Volume 6733 of Lecture Notes in Computer Science* (p. 35-54). Lyngby, Denmark: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-21702-9_3

[32] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., ... Vikkelsoe, C. (2007). PRESENT: An ultra-lightweight block cipher. In P. Paillier & I. Verbauwhede (Eds.), *Proceedings of CHES '07 — The 9th Annual International Workshop on Cryptographic Hardware and Embedded Systems, Volume 4727 of Lecture Notes in Computer Science* (p. 450-466). Vienna, Austria: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-74735-2_31

[33] Bogdanov, A., & Rechberger, C. (2011). A 3-subset meet-in-the-middle attack: Cryptanalysis of the lightweight block cipher KTANTAN. In A. Biryukov, G. Gong, & D. R. Stinson (Eds.), *Proceedings of SAC '10 — The 17th Annual International Workshop on Selected Areas in Cryptography, Volume 6544 of Lecture Notes in Computer Science* (p. 229-240). Waterloo, Ontario, Canada: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-19574-7_16

[34] Brown, L., Pieprzyk, J., & Seberry, J. (1990). LOKI — a cryptographic primitive for authentication and secrecy applications. In J. Seberry & J. Pieprzyk (Eds.), *Advances in Cryptology - Proceedings of AUSCRYPT '90 — The International Conference on Cryptology, Volume 453 of Lecture Notes in Computer Science* (p. 229-236). Sydney, Australia: Springer Berlin Heidelberg. doi: 10.1007/BFb0030364

[35] Chen, S., & Dai, Y. (2011). Related-key amplified boomerang attack on 8-round MISTY1. In L. Chao & H. Wang (Eds.), *Proceedings of CHINACRYPT '11* (p. 7-14). Changsha, Hunan, China: Science Press USA Inc.

[36] Courtois, N., Bard, G. V., & Wagner, D. (2008). Algebraic and slide attacks on Keeloq. In K. Nyberg (Ed.), *Proceedings of FSE '08 — The 15th International Workshop on Fast Software Encryption, Volume 5086 of Lecture Notes in Computer Science* (p. 97-115). Lausanne, Switzerland: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-71039-4_6

[37] CRYPTREC. (2003). Cryptography research and evaluation committees. , *Report 2002*.

[38] Cusick, T. W., & Wood, M. C. (1991). Fast software encryption functions. In A. J. Menezes & S. A. Vanstone (Eds.), *Advances in Cryptology - Proceedings of CRYPTO '90 — The 10th Annual International Cryptology Conference, Volume 537 of Lecture Notes in Computer Science* (p. 546-563). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-38424-3_38

[39] Dai, Y. (2012). Personal communications.

[40] Dai, Y., & Chen, S. (2012). Weak-key class of MISTY1 for related-key differential attack. In M. Y. Chuan-Kun Wu & D. Lin (Eds.), *Proceedings of INSCRYPT '11 — The 7th International Conference on Information Security and Cryptology, Volume 7537 of Lecture Notes in Computer Science* (p. 227-236). Beijing, China: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-34704-7_17

[41] den Boer, B. (1993). A simple and key-economical unconditional authentication scheme. *Journal of Computer Security*, *2*(1), 65-72.

[42] Diffie, W., & Hellman, M. E. (1977). Exhaustive cryptanalysis of the NBS data encryption standard. *Computer*, *10*(6), 74-84. doi: 10.1109/C-M.1977.217750

[43] Dodis, Y., Kiltz, E., Pietrzak, K., & Wichs, D. (2012). Message authentication, revisited. In D. Pointcheva & T. Johansson (Eds.), *Advances in Cryptology - Proceedings of EUROCRYPT '12 — The 31th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Volume 7237 of Lecture Notes in Computer Science* (p. 355-374). Cambridge, UK: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-29011-4_22

[44] Dunkelman, O., & Keller, N. (2008). An improved impossible differential attack on MISTY1. In J. Pieprzyk (Ed.), *Advances in Cryptology - Proceedings of ASIACRYPT '08 — The 14th International Conference on the Theory and Application of Cryptology and Information Security, Volume 5350 of Lecture Notes in Computer Science* (p. 441-454). Melbourne, Australia: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-89255-7_27

[45] Dunkelman, O., Keller, N., & Shamir, A. (2010). A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. In T. Rabin (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '10 — The 30th Annual Cryptology Conference, Volume 6223 of Lecture Notes in Computer Science* (p. 393-410). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-14623-7_21

[46] Dworkin, M. J. (2001). *NIST SP 800-38A Recommendations for block cipher modes of operation, methods and techniques* (Tech. Rep.). Gaithersburg, MD, United States.

[47] Dworkin, M. J. (2006). *NIST SP 800-38D Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC* (Tech. Rep.). Gaithersburg, MD, United States.

[48] Dworkin, M. J. (2007). *NIST SP 800-38D Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC* (Tech. Rep.). Gaithersburg, MD, United States.

[49] Feistel, H. (1970). Cryptographic coding for data-bank privacy. *IBM Research Report RC2827, Yorktown Heights, New York*.

[50] Feistel, H. (1973). Cryptography and computer privacy. *Scientific American*, *228*(5), 15-23.

[51] Ferguson, N. (2005). Authentication weaknesses in GCM. , *Available from http : //csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC – GCM/Ferguson2.pdf*.

[52] Gao, T., Gu, Q., & Chen, Z. (2009). A hyperchaos generated from Chen's system. *Chaos, Solitons & Fractals*, *39*(4), 1849-1855. doi: 10.1016/j.chaos.2007.06.125

[53] Gauravaram, P., & Kelsey, J. (2008). Linear-XOR and additive checksums don't protect Damgård-Merkle hashes from generic attacks. In T. Malkin (Ed.), *Proceedings of CT-RSA '08— The Cryptographers' Track at the RSA Conference 2008, Volume 4964 of Lecture Notes in Computer Science* (p. 36-51). San Francisco, CA, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-79263-5_3

[54] Gauravaram, P., Kelsey, J., Knudsen, L. R., & Thomsen, S. S. (2010). On hash functions using checksums. *International Journal of Information Security*, *9*(2), 137-151. doi: 10.1007/s10207 -009-0100-7

[55] Goldwasser, S., & Micali, S. (2014). Probabilistic encryption. *Journal of Computer and System Sciences*, *28*(2), 270-299. doi: 10.1016/0022-0000(84)90070-9

[56] Halevi, S., & Krawczyk, H. (1997). MMH: Software message authentication in the Gbit/second rates. In E. Biham (Ed.), *Proceedings of FSE '97 — The 4th International Workshop on Fast Software Encryption, Volume 1267 of Lecture Notes in Computer Science* (p. 172-189). Haifa, Israel: Springer Berlin Heidelberg. doi: 10.1007/BFb0052345

[57] Hämäläinen, P., Alho, T., Hännikäinen, M., & Hämäläinen, T. D. (2006). Design and implementation of low-area and low-power AES encryption hardware core. In M. F. Abdollah, E. Corchado, & V. Casola (Eds.), *Proceedings of DSD '06 — The 9th Euromicro Conference on Digital System Design: Architectures, Methods and Tools* (p. 577-583). Dubrovnik, Croatia: IEEE. doi: 10.1109/DSD.2006.40

[58] Handschuh, H., & Preneel, B. (2008). Key-recovery attacks on universal hash function based MAC algorithms. In D. Wagner (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '08 — The 28th Annual Cryptology Conference, Volume 5157 of Lecture Notes in Computer Science* (p. 144-161). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-85174-5_9

[59] Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B.-S., ... Chee, S. (2006). HIGHT: A new block cipher suitable for low-resource device. In L. Goubin & M. Matsui (Eds.), *Proceedings of CHES '06 — The 8th Annual International Workshop on Cryptographic Hardware and Embedded Systems, Volume 4249 of Lecture Notes in Computer Science* (p. 46-59). Yokohama, Japan: Springer Berlin Heidelberg. doi: 10.1007/11894063_4

[60] Hong, S., Kim, J., Lee, S., & Preneel, B. (2005). Related-key rectangle attacks on reduced versions of SHACAL-1 and AES-192. In H. Gilbert & H. Handschuh (Eds.), *Proceedings of FSE '05 — The 12th International Workshop on Fast Software Encryption, Volume 3557 of Lecture Notes in Computer Science* (p. 368-383). Paris, France: Springer Berlin Heidelberg. doi: 10.1007/11502760_25

[61] IEEE. (2008). *IEEE standard for floating-point arithmetic* (IEEE Std No. 754-2008). USA: Institute of Electrical and Electronics Engineers.

[62] Igoe, K., & Solinas, J. (2009, August). *AES Galois counter mode for the secure shell transport layer protocol* (RFC No. 5647). RFC Editor. Internet Requests for Comments. Retrieved from `http://www.rfc-editor.org/rfc/rfc5647.txt`

[63] ISO. (1999). *Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher* (ISO/IEC No. 9797-1:2009). Geneva, Switzerland: International Organisation for Standardisation.

[64] ISO. (2005). *Information technology – Security techniques – Encryption algorithms — Part 3: Block ciphers* (ISO/IEC No. 18033-3:2010). Geneva, Switzerland: International Organisation for Standardisation.

[65] ISO. (2009). *Information technology – Security techniques – Authenticated encryption* (ISO/IEC No. 19772:2009). Geneva, Switzerland: International Organisation for Standardisation.

[66] ISO. (2010). *Information technology – Security techniques – Encryption algorithms — Part 3: Block ciphers* (ISO/IEC No. 18033-3:2010). Geneva, Switzerland: International Organisation for Standardisation.

[67] Isobe, T. (2011). A single-key attack on the full GOST block cipher. In A. Joux (Ed.), *Proceedings of FSE '11 — The 18th International Workshop on Fast Software Encryption, Volume 6733 of Lecture Notes in Computer Science* (p. 290-305). Lyngby, Denmark: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-21702-9_17

[68] Isobe, T. (2013). A single-key attack on the full GOST block cipher. *Journal of Cryptology*, *26*(1), 172-189. doi: 10.1007/s00145-012-9118-5

[69] Iwata, T., & Kurosawa, K. (2003). OMAC: One-key CBC MAC. In T. Johansson (Ed.), *Proceedings of FSE '03 — The 10th International Workshop on Fast Software Encryption, Volume 2887 of Lecture Notes in Computer Science* (p. 129-153). Lund, Sweden: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-39887-5_11

[70] Iwata, T., Ohashi, K., & Minematsu, K. (2012). Breaking and repairing GCM security proofs. In R. Safavi-Naini & R. Canetti (Eds.), *Advances in Cryptology - Proceedings of CRYPTO '12 — The 32th Annual Cryptology Conference, Volume 7417 of Lecture Notes in Computer Science* (p. 31-49). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-32009-5_3

[71] Jia, K., & Li, L. (2012). Improved impossible differential attacks on reduced-round MISTY1. In D. H. Lee & M. Yung (Eds.), *Proceedings of WISA '12 — The 13th International Workshop on Information Security Applications, Volume 7690 of Lecture Notes in Computer Science* (p. 15-27). Jeju Island, Korea: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-35416-8_2

[72] Jia, K., Wang, X., Yuan, Z., & Xu, G. (2009). Distinguishing and second-preimage attacks on CBC-like MACs. In J. A. Garay, A. Miyaji, & A. Otsuka (Eds.), *Proceedings of CANS '09 — The 8th International Conference on Cryptology and Network Security, Volume 5888 of Lecture Notes in Computer Science* (p. 349-361). Kanazawa, Japan: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-10433-6_23

[73] Joux, A. (2006). Authentication failures in NIST version of GCM. , *Available from http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/Joux_comments.pdf*.

[74] Kelsey, J., Kohno, T., & Schneier, B. (2001). Amplified boomerang attacks against reduced-round MARS and Serpent. In J. v. L. Gerhard Goos Juris Hartmanis & B. Schneier (Eds.), *Proceedings of FSE '00 — The 7th International Workshop on Fast Software Encryption, Volume 1978 of Lecture Notes in Computer Science* (p. 75-93). New York, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-44706-7_6

[75] Kelsey, J., Schneier, B., & Wagner, D. (1996). Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In N. Koblitz (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '96 — The 16th Annual Cryptology Conference, Volume 1109 of Lecture Notes in Computer Science* (p. 237-251). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-68697-5_19

[76] Kerckhoffs, A. (1883). La cryptographie militaire. *Journal des Sciences Militaires*, *IX*, 5-83.

[77] Kim, J., Hong, S., & Preneel, B. (2007). Related-key rectangle attacks on reduced AES-192 and AES-256. In A. Biryukov (Ed.), *Proceedings of FSE '07— The 14th International Workshop on Fast Software Encryption, Volume 4593 of Lecture Notes in Computer Science* (p. 225-241). Luxembourg, Luxembourg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-74619-5_15

[78] Kim, J., Hong, S., Preneel, B., Biham, E., Dunkelman, O., & Keller, N. (2012). Related-key boomerang and rectangle attacks: Theory and experimental analysis. *IEEE Transactions on Information Theory*, *58*(7), 4948-4966. doi: 10.1109/TIT.2012.2191655

[79] Kim, J., Kim, G., Hong, S., Lee, S., & Hong, D. (2004). The related-key rectangle attack – application to SHACAL-1. In J. P. Huaxiong Wang & V. Varadharajan (Eds.), *Proceedings of ACISP '04 — The 9th Australasian Conference on Information Security and Privacy, Volume 3108 of Lecture Notes in Computer Science* (p. 123-136). Sydney, Australia: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-27800-9_11

[80] Kim, J., Kim, G., Lee, S., Lim, J., & Song, J. (2005). Related-key attacks on reduced rounds of SHACAL-2. In A. Canteaut & K. Viswanathan (Eds.), *Proceedings of INDOCRYPT '04 — The 5th International Conference on Cryptology, Volume 3348 of Lecture Notes in Computer Science* (p. 175-190). Chennai, India: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-30556-9_15

[81] KISA. (1998). SEED algorithm specification. archive available at www.seed.kisa.or.kr:8080/seed/down/seed_specification_english.pdf.

[82] Knudsen, L. R. (1993). Cryptanalysis of LOKI 91. In J. Seberry & Y. Zheng (Eds.), *Advances in Cryptology - Proceedings of AUSCRYPT '92 — The Workshop on the Theory and Application of Cryptographic Techniques, Volume 718 of Lecture Notes in Computer Science* (p. 196-208). Gold Coast, Queensland, Australia: Springer Berlin Heidelberg. doi: 10.1007/3-540-57220-1_62

[83] Knudsen, L. R. (1995). Truncated and higher order differentials. In B. Preneel (Ed.), *Proceedings of FSE '94 — The Second International Workshop on Fast Software Encryption, Volume 1008 of Lecture Notes in Computer Science* (p. 196-211). Leuven, Belgium: Springer Berlin Heidelberg. doi: 10.1007/3-540-60590-8_16

[84] Knudsen, L. R. (1998). DEAL — a 128-bit block cipher. *Technical Report, Department of Informatics, University of Bergen, Norway*.

[85] Knudsen, L. R., & Robshaw, M. J. B. (2011). *The block cipher companion*. Springer -Verlag Berlin Heidelberg.

[86] Knudsen, L. R., & Wagner, D. (2002). Integral cryptanalysis. In J. Daemen & V. Rijmen (Eds.), *Proceedings of FSE '02 — The 9th International Workshop on Fast Software Encryption, Volume 2365 of Lecture Notes in Computer Science* (p. 112-127). Leuven, Belgium: Springer Berlin Heidelberg. doi: 10.1007/3-540-45661-9_9

[87] Kühn, U. (2001). Cryptanalysis of reduced-round MISTY. In B. Pfitzmann (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '01 — The 24th International Conference on the Theory and Application of Cryptographic Techniques, Volume 2045 of Lecture Notes in Computer Science* (p. 325-339). Innsbruck, Austria: Springer Berlin Heidelberg. doi: 10.1007/3-540-44987-6_20

[88] Kühn, U. (2002). Improved cryptanalysis of MISTY1. In J. Daemen & V. Rijmen (Eds.), *Proceedings of FSE '02 — The 9th International Workshop on Fast Software Encryption, Volume 2365 of Lecture Notes in Computer Science* (p. 61-75). Leuven, Belgium: Springer Berlin Heidelberg. doi: 10.1007/3-540-45661-9_5

[89] Kurosawa, K., & Iwata, T. (2003). TMAC: Two-key CBC MAC. In M. Joye (Ed.), *Proceedings of CT-RSA '03 — The Cryptographers' Track at the RSA Conference 2003, Volume 2612 of Lecture Notes in Computer Science* (p. 33-49). San Francisco, CA, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-36563-X_3

[90] Lai, X. (1994). Higher order derivatives and differential cryptanalysis. In R. E. Blahut, D. J. C. Jr., U. Maurer, & T. Mittelholzer (Eds.), *Communications and Cryptography — Two Sides of One Tapestry, Volume 276 of The Springer International Series in Engineering and Computer Science* (p. 227-233). Springer US. doi: 10.1007/978-1-4615-2694-0_23

[91] Lai, X., Massey, J. L., & Murphy, S. (1991). Markov ciphers and differential cryptanalysis. In D. W. Davies (Ed.), *Advances in Cryptology - Proceedings of EUROCRYPT '91 — The Workshop on the Theory and Application of Cryptographic Techniques, Volume 547 of Lecture Notes in Computer Science* (p. 17-38). Brighton, UK: Springer Berlin Heidelberg. doi: 10.1007/3-540-46416-6_2

[92] Law, L., & Solinas, J. (2007, May). *Suite b cryptographic suites for ipsec* (RFC No. 4869). RFC Editor. Internet Requests for Comments. Retrieved from `http://www.rfc-editor.org/rfc/rfc4869.txt` (`http://www.rfc-editor.org/rfc/rfc4869.txt`)

[93] Lee, C., Kim, J., Sung, J., Hong, S., & Lee, S. (2006). Forgery and key recovery attacks on PMAC and Mitchell's TMAC variant. In L. M. Batten & R. Safavi-Naini (Eds.), *Proceedings of ACISP '06 — The 11th Australasian Conference on Information Security and Privacy, Volume 4058 of Lecture Notes in Computer Science* (p. 421-431). Melbourne, Australia: Springer Berlin Heidelberg. doi: 10.1007/11780656_35

[94] Lee, E., Kim, J., Hong, D., Lee, C., Sung, J., Hong, S., & Lim, J. (2008). Weak-key classes of 7-round MISTY 1 and 2 for related-key amplified boomerang attacks. *IEICE Transactions on on Fundamentals on Fundamentals of Electronics, Communications and Computer Sciences*, *91-A*(2), 642-649.

[95] Lee, H., Yoon, J., & Lee, J. (2005, August). *Addition of SEED cipher suites to transport layer security (TLS)* (RFC No. 4162). RFC Editor. Internet Requests for Comments. Retrieved from `http://www.rfc-editor.org/rfc/rfc4162.txt`

[96] Lee, H., Yoon, J., Lee, S., & Lee, J. (2005, October). *The SEED cipher algorithm and its use with IPsec* (RFC No. 4196). RFC Editor. Internet Requests for Comments. Retrieved from `http://www.rfc-editor.org/rfc/rfc4196.txt`

[97] Li, C., Li, S., Álvarez, G., Chen, G., & Lo, K.-T. (2008). Cryptanalysis of a chaotic block cipher with external key and its improved version. *Chaos, Solitons & Fractals*, *37*(1), 299-307. doi: 10.1016/j.chaos.2006.08.025

[98] Li, C., Li, S., Asim, M., Nunez, J., Álvarez, G., & Chen, G. (2009). On the security defects of an image encryption scheme. *Image and Vision Computing*, *27*(9), 1371-1381. doi: 10.1016/j.imavis.2008.12.008

[99] Li, C., Liu, Y., Xie, T., & Chen, M. Z. Q. (2013). Breaking a novel image encryption algorithm based on improved hyperchaotic sequences. *Nonlinear Dynamics*, *73*(3), 2083-2089. doi: 10.1007/s11071-013-0924-6

[100] Li, C., Zhang, L. Y., Ou, R., Wong, K.-W., & Shu, S. (2012). Breaking a novel colour image encryption algorithm based on chaos. *Nonlinear Dynamics*, *70*(4), 2383-2388. doi: 10.1007/s11071-012-0626-5

[101] Lidl, R., & Niederreiter, H. (1984). *Finite fields* (Vol. 20 of Encyclopedia of mathematics and its applications). Cambridge University Press.

[102] Lu, J. (2008). *Cryptanalysis of block ciphers* (Unpublished doctoral dissertation). University of London, UK.

[103] Lu, J. (2009). Related-key rectangle attack on 36 rounds of the XTEA block cipher. *International Journal of Information Security*, *8*(1), 1-11.

[104] Lu, J., & Kim, J. (2008). Attacking 44 rounds of the SHACAL-2 block cipher using related-key rectangle cryptanalysis. *IEICE Transactions on on Fundamentals on Fundamentals of Electronics, Communications and Computer Sciences*, *91-A*(9), 2588-2596.

[105] Lu, J., Kim, J., Keller, N., & Dunkelman, O. (2006). Related-key rectangle attack on 42-round SHACAL-2. In S. K. Katsikas, J. López, M. Backes, S. Gritzalis, & B. Preneel (Eds.), *Proceedings of ISC '06 — The 9th International Conference on Information Security, Volume 4176 of Lecture Notes in Computer Science* (p. 85-100). Samos Island, Greece: Springer Berlin Heidelberg. doi: 10.1007/11836810_7

[106] Lu, J., Kim, J., Keller, N., & Dunkelman, O. (2008). Improving the efficiency of impossible differential cryptanalysis of reduced Camellia and MISTY1. In T. Malkin (Ed.), *Proceedings of CT-RSA '08 — The Cryptographers' Track at the RSA Conference 2008, Volume 4964 of Lecture Notes in Computer Science* (p. 370-386). San Francisco, CA, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-79263-5_24

[107] Lu, J., Lee, C., & Kim, J. (2006). Related-key attacks on the full-round Cobra-F64a and Cobra-F64b. In R. D. Prisco & M. Yung (Eds.), *Proceedings of SCN '06 — The 5th International Conference on Security and Cryptography for Networks, Volume 4116 of Lecture Notes in Computer Science* (p. 95-110). Maiori, Italy: Springer Berlin Heidelberg. doi: 10.1007/11832072_7

[108] Matsui, M. (1997). New block encryption algorithm MISTY. In E. Biham (Ed.), *Proceedings of FSE '97 — The 4th International Workshop on Fast Software Encryption, Volume 1267 of Lecture Notes in Computer Science* (p. 54-68). Haifa, Israel: Springer Berlin Heidelberg. doi: 10.1007/BFb0052334

[109] May, R. M. (1974). Biological populations with nonoverlapping generations: Stable points, stable cycles, and chaos. *Science*, *186*(4164), 645-647. doi: 10.1126/science.186.4164.645

[110] McGrew, D. (2012). Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64-bit block cipher modes. *Cryptology ePrint Archive*. Retrieved from `http://eprint.iacr.org/2012/623` (http://eprint.iacr.org/2012/623)

[111] McGrew, D., & Viega, J. (2005). The security and performance of the Galois/counter mode (GCM) of operation. In A. Canteaut & K. Viswanathan (Eds.), *Proceedings of INDOCRYPT '04 — The 5th International Conference on Cryptology, Volume 3348 of Lecture Notes in Computer Science* (p. 343-355). Chennai, India: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-30556-9_27

[112] Merkle, R. C. (1990). A fast software one-way hash function. *Journal of Cryptology*, *3*(1), 43-58. doi: 10.1007/BF00203968

[113] Merkle, R. C. (1991). Fast software encryption functions. In A. J. Menezes & S. A. Vanstone (Eds.), *Advances in Cryptology - Proceedings of CRYPTO '90 — The 10th Annual International Cryptology Conference, Volume 537 of Lecture Notes in Computer Science* (p. 477-501). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-38424-3_34

[114] Minematsu, K. (2015). AES-OTR v2. Retrieved from `https://competitions.cr.yp.to/round2/aesotrv2.pdf` (https://competitions.cr.yp.to/round2/aesotrv2.pdf)

[115] Minematsu, K., & Matsushima, T. (2007). New bounds for PMAC, TMAC, and XCBC. In A. Biryukov (Ed.), *Proceedings of FSE '07— The 14th International Workshop on Fast Software Encryption, Volume 4593 of Lecture Notes in Computer Science* (p. 434-451). Luxembourg, Luxembourg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-74619-5_27

[116] Mirzaei, O., Yaghoobi, M., & Irani, H. (2012). A new image encryption method: Parallel sub-image encryption with hyper chaos. *Nonlinear Dynamics*, *67*(1), 557-566. doi: 10.1007/s11071-011-0006-6

[117] Miyaguchi, S., Ohta, K., & Iwata, M. (1990). 128-bit hash function (N-Hash). In *Proceedings of SECURICOM 90'* (p. 123-137).

[118] Murphy, S. (2011). The return of the cryptographic boomerang. *IEEE Transactions on Information Theory*, *57*(4), 2517-2521. doi: 10.1109/TIT.2011.2111091

[119] Nandi, M., & Mandal, A. (2008). Improved security analysis of PMAC. *Journal of Mathematical Cryptology*, *2*(2), 149-162. doi: 10.1515/JMC.2008.007

[120] NBS. (1977). *Data encryption standard (DES)* (FIPS No. 46). U.S.: National Bureau of Standards.

[121] NESSIE. (2004). Final report of European project IST-1999-12324.

[122] NIST. (2001). *Advanced encryption standard (AES)* (FIPS No. 197). U.S.: National Institute of Standards and Technology.

[123] Norouzi, B., & Mirzakuchaki, S. (2014). A fast color image encryption algorithm based on hyper-chaotic systems. *Nonlinear Dynamics*, *78*(2), 995-1015. doi: 10.1007/s11071-014-1492-0

[124] Pareek, N. K., Patidar, V., & Sud, K. K. (2003). Discrete chaotic cryptography using external key. *Physics Letters A*, *309*(1-2), 75-82. doi: 10.1016/S0375-9601(03)00122-1

[125] Pareek, N. K., Patidar, V., & Sud, K. K. (2005). Cryptography using multiple one-dimensional chaotic maps. *Communications in Nonlinear Science and Numerical Simulation*, *10*(7), 715-723. doi: 10.1016/j.cnsns.2004.03.006

[126] Pareek, N. K., Patidar, V., & Sud, K. K. (2006). Image encryption using chaotic logistic map. *Image and Vision Computing*, *24*(9), 926-934. doi: 10.1016/j.imavis.2006.02.021

[127] Pareek, N. K., Patidar, V., & Sud, K. K. (2009). A new substitution-diffusion image cipher using chaotic standard and logistic maps. *Communications in Nonlinear Science and Numerical Simulation*, *14*(7), 3056-3075. doi: 10.1016/j.cnsns.2008.11.005

[128] Pareek, N. K., Patidar, V., & Sud, K. K. (2013). Diffusion-substitution based gray image encryption scheme. *Digital Signal Processing*, *23*(3), 894-901. doi: 10.1016/j.dsp.2013.01.005

[129] Park, J., Lee, S., Kim, J., & Lee, J. (2005, February). *Use of the SEED encryption algorithm in cryptographic message syntax (CMS)* (RFC No. 4010). RFC Editor. Internet Requests for Comments. Retrieved from http://www.rfc-editor.org/rfc/rfc4010.txt

[130] Patidar, V., Pareek, N. K., Purohit, G., & Sud, K. K. (2010). Modified substitution-diffusion image cipher using chaotic standard and logistic maps. *Communications in Nonlinear Science and Numerical Simulation*, *15*(10), 2755-2765. doi: 10.1016/j.cnsns.2009.11.010

[131] Peyravian, M., & Coppersmith, D. (1999). A structured symmetric-key block cipher. *Computers & Security*, *18*(2), 134-147. doi: 10.1016/S0167-4048(99)90053-6

[132] PKCS. (2009). PKCS #11 mechanisms v2.30: Cryptoki - draft 7.

[133] Procter, G., & Cid, C. (2013). On weak keys and forgery attacks against polynomial-based MAC schemes. In S. Moriai (Ed.), *Proceedings of FSE '13 — The 20th International Workshop on Fast Software Encryption, Volume 8424 of Lecture Notes in Computer Science* (p. 287-304). Singapore: Springer Berlin Heidelberg. doi: 10.1007/978-3-662-43933-3_15

[134] Qi, G., Chen, G., Du, S., Chen, Z., & Yuan, Z. (2005). Analysis of a new chaotic system. *Physica A: Statistical Mechanics and its Applications*, *352*(2-4), 295-308. doi: 10.1016/j.physa.2004.12.040

[135] Rogaway, P. (2002). Authenticated-encryption with associated-data. In V. Atluri (Ed.), *Proceedings of CCS '02 — The 9th ACM Conference on Computer and Communications Security* (p. 98-107). Washington, DC, USA: ACM New York, NY, USA. doi: 10.1145/586110.586125

[136] Rogaway, P. (2004). Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In P. J. Lee (Ed.), *Advances in Cryptology - Proceedings of ASIACRYPT '04 — The 10th International Conference on the Theory and Application of Cryptology and Information Security, Volume 3329 of Lecture Notes in Computer Science* (p. 16-31). Jeju Island, Korea: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-30539-2_2

[137] Saarinen, M.-J. O. (2012). Cycling attacks on GCM, GHASH and other polynomial MACs and hashes. In A. Canteaut (Ed.), *Proceedings of FSE '12 — The 19th International Workshop on Fast Software Encryption, Volume 7549 of Lecture Notes in Computer Science* (p. 216-225). Washington, DC, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-34047-5_13

[138] Salter, M., Rescorla, E., & Housley, R. (2009, March). *Suite B profile for transport layer security (TLS)* (RFC No. 5430). RFC Editor. Internet Requests for Comments. Retrieved from http://www.rfc-editor.org/rfc/rfc5430.txt (http://www.rfc-editor.org/rfc/rfc5430.txt)

[139] Sarkar, P. (2011). A trade-off between collision probability and key size in universal hashing using polynomials. *Designs, Codes and Cryptography*, *58*(3), 271-278. doi: 10.1007/s10623-010-9408-6

[140] Selçuk, A. A. (2008). On probability of success in linear and differential cryptanalysis. *Journal of Cryptology*, *21*(1), 131-147. doi: 10.1007/s00145-007-9013-7

[141] Shannon, C. E. (1949). Communication theory of secrecy systems. *Bell System Technical Journal*, *28*(4), 656-715.

[142] Shimizu, A., & Miyaguchi, S. (1988). Fast data encipherment algorithm FEAL. In D. Chaum & W. L. Price (Eds.), *Advances in Cryptology - Proceedings of EUROCRYPT '91 — The Annual International Workshop on the Theory and Application of of Cryptographic Techniques, Volume304 of Lecture Notes in Computer Science* (p. 267-278). Amsterdam, The Netherlands: Springer Berlin Heidelberg. doi: 10.1007/3-540-39118-5_24

[143] Sun, X., & Lai, X. (2009). Improved integral attacks on MISTY1. In V. R. Michael J. Jacobson Jr. & R. Safavi-Naini (Eds.), *Proceedings of SAC '09 — The 16th Annual International Workshop on Selected Areas in Cryptography, Volume 5867 of Lecture Notes in Computer Science* (p. 266-280). Calgary, Alberta, Canada: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-05445-7_17

[144] Sung, J. (2011). Differential cryptanalysis of eight-round SEED. *Information Processing Letters*, *111*(10), 474-478. doi: 10.1016/j.ipl.2011.02.004

[145] Tanaka, H., Hatano, Y., Sugio, N., & Kaneko, T. (2007). Security analysis of MISTY1. In S. Kim, M. Yung, & H.-W. Lee (Eds.), *Proceedings of WISA '07 — The 8th International Workshop on Information Security Applications, Volume 4867 of Lecture Notes in Computer Science* (p. 215-226). Jeju Island, Korea: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-77535-5_16

[146] Taylor, R. (1994). An integrity check value algorithm for stream ciphers. In D. R. Stinson (Ed.), *Advances in Cryptology - Proceedings of CRYPTO '93 — The 13th Annual International Cryptology Conference, Volume 773 of Lecture Notes in Computer Science* (p. 40-48). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/3-540-48329-2_4

[147] Todo, Y. (2015). Integral cryptanalysis on full MISTY1. In R. Gennaro & M. Robshaw (Eds.), *Advances in Cryptology - Proceedings of CRYPTO '96 — The 16th Annual International Cryptology Conference, Volume 9215 of Lecture Notes in Computer Science* (p. 413-432). Santa Barbara, California, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-662-47989-6_20

[148] Tong, X.-J., Wang, Z., Zhang, M., & Liu, Y. (2013). A new algorithm of the combination of image compression and encryption technology based on cross chaotic map. *Nonlinear Dynamics*, *72*(1-2), 229-241. doi: 10.1007/s11071-012-0707-5

[149] Tsunoo, Y., Saito, T., Nakashima, H., & Shigeri, M. (2009a). Higher order differential attack on 6-round MISTY1. *IEICE Transactions on on Fundamentals on Fundamentals of Electronics, Communications and Computer Sciences*, *92-A*(1), 3-10.

[150] Tsunoo, Y., Saito, T., Nakashima, H., & Shigeri, M. (2009b). Higher order differential attacks on reduced-round MISTY1. In P. J. Lee & J. H. Cheon (Eds.), *Proceedings of ICISC '08 — The 11th International Conference on Information Security and Cryptology, Volume 5461 of Lecture Notes in Computer Science* (p. 415-431). Seoul, Korea: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-00730-9_26

[151] Tsunoo, Y., Saito, T., Nakashima, H., & Shigeri, M. (2010). Security analysis of 7-round MISTY1 against higher order differential attacks. *IEICE Transactions on on Fundamentals on Fundamentals of Electronics, Communications and Computer Sciences*, *93-A*(1), 144-152.

[152] Tsunoo, Y., Saito, T., Nakashima, H., & Shigeri, M. (2012). Finding higher order differentials of MISTY1. *IEICE Transactions on on Fundamentals on Fundamentals of Electronics, Communications and Computer Sciences*, *95-A*(6), 1049-1055.

[153] TTA. (1999). 128-bit block cipher SEED, TTAS.KO-12.2004.

[154] Wagner, D. (1999). The boomerang attack. In L. R. Knudsen (Ed.), *Proceedings of FSE '99 — The 4th International Workshop on Fast Software Encryption, Volume 1636 of Lecture Notes in Computer Science* (p. 156-170). Rome, Italy: Springer Berlin Heidelberg. doi: 10.1007/3-540-48519-8_12

[155] Wang, L. (2015). SHELL v2.0. Retrieved from `https://competitions.cr.yp.to/round2/shellv20.pdf` (https://competitions.cr.yp.to/round2/shellv20.pdf)

[156] Wang, X., & Guo, K. (2014). A new image alternate encryption algorithm based on chaotic map. *Nonlinear Dynamics*, *76*(4), 1943-1950. doi: 10.1007/s11071-014-1259-7

[157] Wang, X., & Liu, L. (2013). Cryptanalysis of a parallel sub-image encryption method with high-dimensional chaos. *Nonlinear Dynamics*, *73*(1-2), 795-800. doi: 10.1007/s11071-013-0832-9

[158] Wang, X., & Wang, Q. (2014). A novel image encryption algorithm based on dynamic s-boxes constructed by chaos. *Nonlinear Dynamics*, *75*(3), 567-576. doi: 10.1007/s11071-013-1086-2

[159] Wegman, M. N., & Carter, J. L. (1981). New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, *22*(3), 265-279. doi: 10.1016/0022-0000(81)90033-7

[160] Wei, J., Liao, X., wo Wong, K., & Zhou, T. (2007). Cryptanalysis of a cryptosystem using multiple one-dimensional chaotic maps. *Communications in Nonlinear Science and Numerical Simulation*, *12*(5), 814-822. doi: 10.1016/j.cnsns.2005.06.001

[161] Yanami, H., & Shimoyama, T. (2003). Differential cryptanalysis of a reduced-round SEED. In S. Cimato, G. Persiano, & C. Galdi (Eds.), *Proceedings of SCN '02 — The 3th International Conference on Security and Cryptography for Networks, Volume 2576 of Lecture Notes in Computer Science* (p. 186-198). Amalfi, Italy: Springer Berlin Heidelberg. doi: 10.1007/3-540-36413-7_14

[162] Yoon, S., Kim, J., Park, H., Jeong, H., & Won, Y. (2010, August). *The SEED cipher algorithm and its use with the secure real-time transport protocol (SRTP)* (RFC No. 5669). RFC Editor. Internet Requests for Comments. Retrieved from `http://www.rfc-editor.org/rfc/rfc5669.txt`

[163] Yujun, N., Xingyuan, W., Mingjun, W., & Huaguang, Z. (2010). A new hyperchaotic system and its circuit implementation. *Communications in Nonlinear Science and Numerical Simulation*, *15*(11), 3518-3524. doi: 10.1016/j.cnsns.2009.12.005

[164] Zhang, Y.-Q., & Wang, X.-Y. (2014). Analysis and improvement of a chaos-based symmetric image encryption scheme using a bit-level permutation. *Nonlinear Dynamics*, *77*(3), 687-698. doi: 10.1007/s11071-014-1331-3

LIST OF PUBLICATIONS

Journal Articles

[1] Lu, J., Yap, W.-S., Henricksen, M., & Heng, S.-H. (2014). Differential attack on nine rounds of the SEED block cipher. *Information Processing Letters*, *114*(3), 116-123. doi: 10.1016/j.ipl.2013.11.004

[2] Yap, W.-S., Phan, R. C.-W., Yau, W.-C., & Heng, S.-H. (2015). Cryptanalysis of a new image alternate encryption algorithm based on chaotic map. *Nonlinear Dynamics*, *80*(3), 1483-1491. doi: 10.1007/s11071-015-1956-x

[3] Yap, W.-S., Yeo, S. L., Heng, S.-H., & Henricksen, M. (2014a). Parallelizable MAC revisited. *Security and Communication Networks*, *7*(7), 1115-1127. doi: 10.1002/sec.842

[4] Yap, W.-S., Yeo, S. L., Heng, S.-H., & Henricksen, M. (2014b). Security analysis of GCM for communication. *Security and Communication Networks*, *7*(5), 854-864. doi: 10.1002/sec.798

Conference Proceedings

[1] Lu, J., Yap, W.-S., & Wei, Y. (2013). Weak keys of the full MISTY1 block cipher for related-key differential cryptanalysis. In E. Dawson (Ed.), *Proceedings of CT-RSA '13 — The Cryptographers' Track at the RSA Conference 2013, Volume 7779 of Lecture Notes in Computer Science* (p. 389-404). San Francisco, CA, USA: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-36095-4_25

[2] Yap, W.-S., Yeo, S. L., & Yian, C. H. (2011). Cryptanalysis of the full CHAIN cipher. In M. F. Abdollah, E. Corchado, & V. Casola (Eds.), *Proceedings of IAS '11 — The 7th International Conference on Information Assurance and Security* (p. 40-45). Melaka, Malaysia: IEEE. doi: 10.1109/ISIAS.2011.6122792

Manuscripts

[1] Yap, W.-S., Phan, R. C.-W., Goi, B.-M., Yau, W.-C., & Heng, S.-H. (2015). On the effective key space of recent image encryption algorithms using external key. *In preparation*.