RELATED-KEY ATTACKS ON BLOCK CIPHERS

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

BY

ASLI DARBUKA

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

AUGUST 2009

Approval of the thesis:

RELATED-KEY ATTACKS ON BLOCK CIPHERS

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

Prof. Dr. Ersan AKYILDIZ _____
Director, Graduate School of **Applied Mathematics**

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

Assoc. Prof. Dr. Ali DOĞANAKSOY _____
Supervisor, **Department of Mathematics, METU**

Examining Committee Members:

Prof. Dr. Ersan AKYILDIZ _____
Department of Mathematics, METU

Assoc. Prof. Dr. Ali DOĞANAKSOY _____
Department of Mathematics, METU

Assist. Prof. Dr. Zülfükar SAYGI _____
Department of Mathematics, TOBB ETÜ

Dr. Muhiddin UĞUZ _____
Department of Mathematics, METU

Dr. Meltem Sönmez TURAN _____
Institute of Applied Mathematics, METU

Date: _____

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

Name, Last Name: ASLI DARBUKA

Signature :

ABSTRACT

RELATED-KEY ATTACKS ON BLOCK CIPHERS

Darbuka, Aslı
M.S., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Ali DOĞANAKSOY

August 2009, 123 pages

One of the most important cryptographic primitives is the concept of block ciphers which yields confidentiality for data transmission in communication. Therefore, to be sure that confidentiality is provided, it is necessary to analyse the security of block ciphers by investigating their resistance to existing attacks. For this reason, related-key attacks gain much popularity in recent years and have been applied to many block ciphers with weak key schedules. In this work, our main motivation is to cover types of related-key attacks on block ciphers and exemplify them.

For years, cryptanalysts have been investigating the security of the block cipher XTEA and proposed several attacks on the cipher. First in FSE'02, Moon et al. presented a 14-round impossible differential attack on XTEA. Then in ICISC'03, Hong et al. proposed a 15-round differential attack and a 23-round truncated differential attack on XTEA. In FSE'04, Ko et al. proposed a 27-round related-key truncated differential attack on XTEA. Afterwards, in Vietcrypt'06, Lee et al. proposed a 34-round related-key rectangle attack on XTEA. Finally in 2008, Lu improved this attack to a related-key rectangle attack on 36-round XTEA which

is the best attack on XTEA in terms of the number of attacked rounds. In this thesis, we also analyse differential properties of both structure and key schedule of XTEA block cipher and introduce our 25-round related-key impossible differential distinguisher for XTEA.

Keywords: Block Ciphers, Cryptanalysis, Related-Key Attacks, XTEA

ÖZ

BLOK ŞİFRELERE YAPILAN İLİŞİK ANAHTAR ATAKLARI

Darbuka, Aslı
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali DOĞANAKSOY

Ağustos 2009, 123 sayfa

Blok şifreler kriptografinin en önemli yapıtaşlarından biri olup haberleşme esnasında veri aktarımının gizliliğini sağlar. Bu nedenle, gizliliğin sağlandığından emin olmak için, blok şifrelerin güvenliğinin varolan ataklara olan dayanıklılığı araştırılarak analiz edilmesi gerekir. Bu sebeple, son yıllarda ilişik anahtar atakları önemli ölçüde yaygınlaştı ve birçok zayıf anahtar üreteçli blok şifreye uygulandı. Bu çalışmadaki temel amacımız, blok şifrelere yapılan ilişik anahtar atak çeşitlerini ele almak ve onları örneklendirmektir.

Yıllardır, kriptanalistler XTEA blok şifresinin güvenliğini araştırıyorlar ve bu şifreye yapılmış birçok atak sundular. İlk olarak FSE'02'de, Moon vd 14 çevrimlik olanaksız diferansiyel atağını sundular. Sonra ICISC'03'te, Hong vd 15 çevrimlik diferansiyel atak ve 23 çevrimlik kesik diferansiyel atak sundular. FSE'04'te, Ko vd 27 çevrimlik ilişik anahtarlı kesik diferansiyel atak sundular. Daha sonra Vietcrypt'06'da, Lee vd 34 çevrimlik ilişik anahtarlı dikdörtgen atağını sundular. Son olarak da 2008'de bu atak Lu tarafından çevrim sayısı bakımından XTEA'ye yapılan en iyi atak olan 36 çevrimlik ilişik anahtarlı dikdörtgen atağına geliştirilmiştir. Bu tezde ayrıca XTEA'nin yapısının ve anahtar üretecinin diferansiyel özelliğini inceledik ve XTEA için 25 çevrimlik ilişik anahtarlı olanaksız diferansiyel ayırıcı sunduk.

Anahtar Kelimeler: Blok Şifreler, Kriptanaliz, İlişik Anahtar Atakları, XTEA

To my family

ACKNOWLEDGMENTS

First of all, I am most grateful to my supervisor, Assoc. Prof. Dr. Ali Doğanaksoy, for his guidance and attention throughout my studies at METU. He has always been a driving force and a source of inspiration.

Several people assisted me in my work. I would like to express my gratitude to Dr. Meltem Sönmez Turan who has been a great collaborator and a supporter. She was there every time I needed her opinion and advice, always patient and good-willed. I also wish to thank Fatih Sulak for discussions over my work. His comments and constructive criticism were valuable. My colleagues, Neşe Öztop, Dilek Özberk, Onur Koçak, Onur Özen, İ.Firuze Atalay and Kerem Varıcı deserve special thanks for their moral support, presence and humour. Above all, I want to thank M. Fatih Bay for his special love and support.

Last, but not least, I wish to thank my parents, Ahmet and Semra, for raising me to become who I am and for always being a great support.

TABLE OF CONTENTS

ABSTRACT iv

ÖZ vi

DEDICATION viii

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xv

LIST OF FIGURES xvi

CHAPTERS

1 INTRODUCTION 1

1.1 Block Ciphers 2

1.2 Differential Cryptanalysis 4

1.3 Linear Cryptanalysis 7

1.4 The Structure of Thesis and Our Contributions 9

2 RELATED-KEY CRYPTANALYSIS 11

2.1 The Chosen-Key Attack of Winternitz and Hellman 12

2.2 Conventional Related-Key Attacks 13

2.2.1 Related-Key Known Plaintext Attack 13

2.2.2 Conventional Related-Key Attacks on Ciphers with Feistel Structure: 14

2.3 A Related-Key Attack on LOKI89 17

2.3.1 A Related-Key Attack on Block Ciphers with More Generalized Key Schedule 19

2.4 A Related-Key Attack on full round LOKI91 21

2.5 Related-Key Cryptanalysis of full-round SHACAL-1 22

2.5.1 An Attack on full-round SHACAL-1 23

3 RELATED-KEY DIFFERENTIAL CRYPTANALYSIS 25

3.1 Related-Key Differential Cryptanalysis of GOST 25

3.1.1 Notations 26

3.1.2 The GOST Block Cipher 26

3.1.2.1 Key Scheduling Algorithm 26

3.1.2.2 Encryption Algorithm 27

3.1.3 A Distinguishing Attack on Full Round GOST 28

3.1.4 A Related-Key Differential Attack on Full Round GOST . 28

3.1.4.1 A 30-Round Related-Key Differential Characteristics for GOST 28

3.1.4.2 A Full Round Attack on GOST 29

3.2 Related-Key Differential Cryptanalysis of KASUMI 30

3.2.1 Notations 30

3.2.2 The Block Cipher KASUMI 31

3.2.2.1 The Key Scheduling Algorithm of KASUMI . 31

3.2.2.2 The Encryption Algorithm of KASUMI . . . 32

3.2.3 Related-Key Differential Attack on 5 and 6 Rounds KASUMI 34

3.2.3.1 A Set of 4-Round Differentials for KASUMI . 34

3.2.3.2 An Attack on 5-Round KASUMI 36

3.2.3.3 An Attack on 6-Round KASUMI 38

4 RELATED-KEY IMPOSSIBLE DIFFERENTIAL CRYPTANALYSIS 40

4.1 Overview of the Attack 41

4.2 Related-Key Impossible Differential Cryptanalysis of Reduced-Round AES 42

4.2.1 Notations 43

4.2.2 The AES Block Cipher 43

4.2.3 Some Properties of AES-192 45

4.2.4 A 5.5-Round Related Key Impossible Differential of AES-192 46

4.2.5 A 7-Round Related-Key Impossible Differential Attack on AES-192 . . . 49
4.2.6 Attack Complexity . . . 51
4.3 Related-Key Impossible Differential Cryptanalysis of 31-Round HIGHT 52
4.3.1 Notations . . . 52
4.3.2 The HIGHT Block Cipher . . . 53
4.3.2.1 Key Scheduling Algorithm . . . 53
4.3.2.2 The Encryption Function of HIGHT . . . 54
4.3.3 A 31-Round Related-Key Impossible Differential Attack on HIGHT . . . 55
4.3.3.1 A 22-Round Related-Key Impossible Differential for HIGHT . . . 57
4.3.3.2 The Attack . . . 57
4.4 Our Related-Key Impossible Differential Distinguisher for XTEA . . 62
4.4.1 Notations . . . 63
4.4.2 The XTEA Block Cipher . . . 63
4.4.2.1 The Key Schedule of XTEA . . . 63
4.4.2.2 The Encryption Function of XTEA . . . 63
4.4.3 A 12-Round Impossible Differential by Moon et al. . . . 64
4.4.4 Our 25-Round Related-Key Impossible Distinguisher . . . 65
5 RELATED-KEY BOOMERANG ATTACK AND ITS EXTENSIONS . . . 68
5.1 Related-Key Boomerang Attack . . . 68
5.1.1 Related-Key Boomerang Attack on Reduced-Round IDEA 70
5.1.1.1 Notations . . . 71
5.1.1.2 The IDEA Block Cipher . . . 71
5.1.1.3 A Related-Key Boomerang Attack 6-Round IDEA . . . 73
5.2 Related-Key Amplified Boomerang Attack . . . 76
5.3 Related-Key Rectangle Attack . . . 78
5.3.1 A Related-Key Rectangle Attack on the Full Round SHACAL-1 . . . 80
5.3.1.1 Notations . . . 80

5.3.1.2 The SHACAL-1 Block Cipher 80
5.3.1.3 Related-Key Rectangle Attack on Full Round SHACAL-1 82
5.3.1.4 A 69-Round Related-Key Rectangle Distinguisher for SHACAL-1 82
5.3.1.5 The Attack 83
5.4 Related-Key Impossible Boomerang Attack 87
5.4.1 A 6-Round Related-Key Impossible Boomerang Distinguisher for AES-192 89
6 RELATED-KEY DIFFERENTIAL-LINEAR CRYPTANALYSIS 92
6.1 Overview of the Attack 92
6.2 Related-Key Differential-Linear Cryptanalysis of Reduced-Round AES-192 94
6.2.1 Notations 94
6.2.2 A 5-Round Related-Key Differential-Linear Distinguisher for AES-192 94
6.2.3 A 7-Round Related Key Differential-Linear Attack on AES-192 97
6.2.4 Attack Complexity 97
7 SLIDE ATTACKS 99
7.1 Slide Attack 99
7.1.1 A Typical Slide Attack 100
7.1.2 Slide Attack on Feistel Ciphers 101
7.2 Advanced Slide Techniques 103
7.2.1 Complementation Slide 103
7.2.2 Sliding with a Twist 105
7.2.2.1 Cryptanalysis of DESX with Sliding Twist Technique 106
7.2.3 Realigning Slide 107
7.2.3.1 Realigning Slide Attack on Full Round DES . 107
7.2.4 Methods for Handling Stronger Functions 110
7.3 Improved Slide Attacks 111
7.3.1 Improved Slide Technique 111

7.3.2 Improved Slide Attack on 24-Round GOST 113
8 CONCLUSION 116
REFERENCES 118

LIST OF TABLES

TABLES

Table 3.1 The Key Schedule of GOST 27

Table 3.2 A 32-Round Related-Key Differential Characteristic for GOST 28

Table 3.3 A 30-Round Related-Key Differential Characteristic for GOST 29

Table 3.4 The Key Schedule of KASUMI 31

Table 3.5 Constants Used in the Key Schedule of KASUMI 31

Table 4.1 Subkey Differences 46

Table 4.2 Relations Between the Master Key, Whitening Keys and Round Keys 56

Table 4.3 A 22-Round Related-Key Impossible Differential for HIGHT 57

Table 4.4 The Key Schedule of XTEA 64

Table 4.5 A 12-Round Impossible Differential Distinguisher for XTEA 65

Table 4.6 Our 25-Round Related-Key Impossible Differential Characteristic for XTEA 65

Table 5.1 Key Schedule of IDEA 72

Table 5.2 The First Related-Key Differential for SHACAL-1 85

Table 5.3 The Second Related-Key Differential for SHACAL-1 86

Table 5.4 Subkey Differences 89

Table 6.1 Subkey Differences 95

Table 7.1 Circular Shifts in the Tweaked Key Schedule of DES 108

Table 7.2 A 7-Round Differential Characteristics with Probability of 0.494 for GOST 114

LIST OF FIGURES

FIGURES

Figure 2.1 Conventional related-key attack 14
Figure 2.2 Conventional related-key attack on a generic Feistel cipher 15
Figure 2.3 Related-key attack on full-round LOKI89 20

Figure 3.1 ith round function of GOST 27
Figure 3.2 Functions of KASUMI 34
Figure 3.3 A set of 4-round related-key differential characteristics for KASUMI . . . 36

Figure 4.1 A 4.5-round related-key differential for AES-192 47
Figure 4.2 A 1-round related-key differential for AES-192 48
Figure 4.3 A 5.5 round related-key impossible differential distinguisher for AES-192 . 48
Figure 4.4 Rounds before and after the related-key impossible distinguisher of AES-192 49
Figure 4.5 ith round function of HIGHT 55
Figure 4.6 ith round function of XTEA 64

Figure 5.1 Related-key boomerang distinguisher based on four related keys 69
Figure 5.2 ith round of IDEA 72
Figure 5.3 The first related-key differential is on the left-hand side and the second related-key differential is on the right-hand side for IDEA 74
Figure 5.4 Related-key amplified boomerang distinguisher based on four related keys 77
Figure 5.5 Related-key rectangle distinguisher based on four related keys 79
Figure 5.6 ith round function of SHACAL-1 82
Figure 5.7 Related-key impossible boomerang distinguisher satisfying the condition $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$ 88

Figure 5.8 Differentials with probability 1 for AES-192 90

Figure 6.1 A 4-round differential for AES 96

Figure 7.1 Typical slide attack 100

Figure 7.2 Typical slide attack on a generic *r*-round Feistel cipher with one round self-similarity 102

Figure 7.3 Complementation slide attack on a generic *r*-round Feistel cipher with two-round self-similarity 104

Figure 7.4 Sliding with a twist attack on a generic Feistel cipher with two-round self-similarity 105

Figure 7.5 Twisting slide attack on full-round DESX 106

Figure 7.6 Shift pattern 108

CHAPTER 1

INTRODUCTION

Cryptology which comes from Greek words *kryptós* “hidden”, *grápho* “write” or *legen* “to speak” is the study of hiding sensitive information. It incorporates two subjects, namely *cryptography* and *cryptanalysis*. Cryptography is a tool to protect informations and to design secure cryptographic algorithms and cryptanalysis on the other hand is the study of getting meaning of the encrypted information or breaking cryptographic primitives.

The history of cryptology dates back 4000 years ago. Throughout the ages, people always needed to conceal information due to several reasons. Thus, cryptology was developed and started to become a crucial tool in real-world applications since then. For example, during World War I and II, cryptanalysis played an important role that secret information was cryptanalyzed by the adverse party changed the course of events.

After the development of computers and communication sciences in 1960s, the need of secrecy emerged and cryptology had been become famous. More explicitly, private sector wanted to keep their digital information secret. For this reason in early 1970s, IBM designed DES (Data Encryption Standard) as U.S Federal Information Processing Standard (FIPS) which is the building structure of most of the cryptographic algorithms. It had been used as the standard in many financial institutions all over the world until it was broken.

The most common property of the algorithms designed until that time is that they use only one secret parameter- namely *secret key*. That is, this secret key is shared by the two parties who are communicating and making encryption and decryption. Such algorithms are called

Symmetric Key Algorithms which are commonly used in many world-wide applications.

Afterwards, in 1976, Diffie and Hellman introduced a new concept *Public-Key Cryptography* which is a type of *Asymmetric Key Algorithms*. In this technique, a pair of keys are used- *a public key* used for encryption and *a private key* used for decryption. Then in 1978, the idea of using public-key was firstly used by Rivest, Shamir and Aleman and they proposed a practical public-key encryption and signature scheme called RSA which is based on hardness of factoring large integers. Then in 1985, El Gamal proposed another public-key cryptosystem called "El Gamal encryption system" based on the discrete logarithm problem. Subsequently in 1991, one of the most striking development called 'Digital Signature' was presented.
To sum up, recently the development of computer sciences results in increasing of the importance of cryptology on human interest. Cryptology also started to take part in people's activities for secure personal information besides governments' activities. Cryptology is used in many applications such as communications in army, financial transactions, authentication, e-government applications, etc. Then it is expected that day after day, the importance of cryptology have further increased and will become one of the most essential part of the human life.

Next, I am going to describe block ciphers which is an important class of symmetric cryptosystems.

1.1 Block Ciphers

A *block cipher* is a symmetric-key algorithm which translates n-bit data block broken from m-bit block into n-bit encrypted data block by using k-bit secret key. It consists of three building structures: Encryption, Decryption and Key Scheduling Algorithms. Both encryption and decryption algorithms accept an n-bit data block (plaintexts and ciphertexts, respectively) and a k-bit secret key as input and generates n-bit output block. In block ciphers, decryption is the inverse function of encryption. Generally, a block cipher is a permutation for each key over input data block. In addition, it is commonly agreed that an ideal block cipher can be seen as the set of all secret key combinations which is indistinguishable from a random permutation. The types of block ciphers can be classified into two types of structures: Feistel Networks and

Substitution-Permutation Networks (SPN).

Feistel Networks: In Feistel Networks, the input data is split into two halves and in each round, round function is applied to only one half and the output of round function is bitwise XORed with the other half. Then, two halves are swapped that becomes as an input to the next round function. Data Encryption Standard (DES) is an example of Feistel Networks.

Substitution-Permutation Networks (SPNs): In SPNs, the round function is applied to complete block of input data. The Advanced Encryption Standard (AES) is an example of SPN.

The main difference between two networks is that Feistel Networks do not need to have bijective round function, but SPNs should have bijective round function to allow unique decryption.

Most of the block ciphers are constructed by iterating a function a number of times. This function is called *a round function* which accepts the output of the previous round and a round key as an input and translates them into output block. In a typical block cipher, a round function consists of a bitwise XOR of round key, non-linear substitution boxes called S-boxes which is a nonlinear function operating on fixed length blocks and Permutations.

There are mainly five types of cryptanalysis scenarios. But, before mentioning cryptanalytic techniques, it is better to know the resources of a cryptanalysist. In cryptography, it is a common agreement that a cryptosystem should satisfy Kerkhoff's Principle given as follows:

Kerkhoff's Principle*: A cryptosystem should be secure even if everything about the system, except the secret key, is public knowledge.*

More explicitly, Kerkhoff states that the secret parameters should be only the secret key and the security of the cryptosystem should rely on only the secret key.

Now, cryptanalytic techniques are mentioned as below:

- *Ciphertext Only Attacks:* In this attack scenario, it is assumed that the attacker has access to a set of ciphertexts and limited knowledge of plaintexts.

- *Known Plaintext Attacks:* In this case, the attacker is assumed to have an access to a set of plaintexts and their corresponding ciphertexts under a fixed secret key. In the attack, the attacker uses known plaintext-ciphertext pairs to derive secret key. Linear cryptanalysis is an example of known plaintext attacks.

- *Chosen Plaintext-Ciphertext Attacks:* In this attack scenario, it is assumed that the attacker can make encryption of a set of plaintexts under the secret key. In other words, he can get the ciphertexts of chosen plaintexts under secret key. Differential cryptanalysis is an example of chosen plaintext-ciphertext attack.

- *Adaptively Chosen Plaintext-Ciphertext Attacks:* In adaptively chosen plaintext-ciphertext attack scenario, the attacker has a power of making encryption/decryption under unknown key and using the information obtained, the attacker can choose another set of plaintext-ciphertext under secret key.

- *Related-Key Attacks:* The attack has a strong assumption that the attacker is able to make encryption under secret key and a key derived from the secret key. Namely, the attacker can choose the keys and there are more than one keys to make encryptions.

Note that Brute Force attack or exhaustive search is a basic and trivial search routine such that the attacker tries all possible secret key combinations to find the secret key. In cryptanalysis, breaking a cipher has a meaning that the attacker distinguishes the cipher from a random permutation and obtains the secret key faster than exhaustive search.

1.2 Differential Cryptanalysis

Differential cryptanalysis was proposed by Biham and Shamir in 1990 [1] and it is a statistical method which is applicable to many block ciphers. The attack was firstly applied to reduced-round DES. Then, the attack on DES was improved and broke full-round DES theoretically

[2, 3].

Differential cryptanalysis is a powerful technique that can break many ciphers, successfully. When differential cryptanalysis was first introduced, many block ciphers were vulnerable to this technique. However, recently designed block ciphers are proved to be secure against differential cryptanalysis.

Afterwards, the extensions of differential cryptanalysis have been proposed: impossible differential attack, boomerang attack, amplified boomerang attack, rectangle attack, differential-linear attack. In the following chapters, we will mention the related-key versions of these attacks.

Differential cryptanalysis exploits specific relations between two plaintexts P and P^* under secret key K. More explicitly, given difference between P and P^* under K, the attacker searches some specific difference between C and C^* with high probability, where C and C^* are corresponding ciphertexts of P and P^* encrypted under secret key K, respectively.

The difference operation can be defined in many different ways, however, from now on, we consider the operation as XOR operation which is the most commonly used operation in differential cryptanalysis. The difference between plaintexts is denoted as the input difference and the difference between ciphertexts is denoted as output difference. A *Differential* is defined as the combination between input and output differences. Namely, A differential $(\alpha \rightarrow \beta)$ with probability p is the prediction that given the input difference α leads to the output difference β with probability p.

Probability of differential for a block cipher is defined as follows:

Definition 1.2.1 *Let E be a block cipher with n-bit block size and k-bit secret key K. Let α and β be input and output difference for E, respectively, then the probability p of the differential $(\alpha \rightarrow \beta)$ is*

$$Pr(\alpha \rightarrow \beta) = Pr[E(P) \oplus E(P \oplus \alpha) = \beta] \tag{1.1}$$

Moreover, the probability of the differential is computed as

$$p = Pr(\alpha \rightarrow \beta) = \frac{\sharp\{P \,|E(P) \oplus E(P \oplus \alpha) = \beta\}}{2^n}. \tag{1.2}$$

For a random permutation with n-bit block size, it is expected to get the corresponding difference occurs with probability 2^{-n}. However, in differential cryptanalysis, the attacker tries to find a differential that has a probability much higher than 2^{-n} to mount the attack.

There are many different intermediate differences that leads to same differential. Therefore, a new notion was introduced: A *differential characteristics* is defined as a sequence of an intermediate differences which leads to a particular differential. A differential characteristics for r-round with probability p is called r-round differential characteristics.

In most block ciphers, given input difference propagates linearly in all components of round function except S-boxes. In a typical round function, the only nonlinear component is S-boxes which controls the difference in nonlinear way. For this reason, *Difference Distribution Table (DDT)* or the XOR table is defined to analyse the probability of differences by counting the number of pairs with the given input difference leads to the given input differences.

For the key recovery attack, the the attacker aims to take advantage of desired output differences of the rounds before the last one or two rounds. More explicitly, 1 or 2 rounds are added to the end of the differential characteristics depending on the structure of the cipher(SPN or Feistel Network). Then, the attacker guesses necessary key bits/bytes, decrypts all ciphertext pairs until the end of differential characteristics and checks the pairs that the characteristics hold with the given probability.

The number of data needed to implement the attack depends on the probability of the differential characteristics, number of right pairs needed and the number of guessed subkey bits.

A pair of plaintexts that satisfy the given plaintext-ciphertext differences and follow the differential characteristics at the inner rounds is called *a right pair* that suggests a set of right keys. Sometimes, a pair of plaintexts can satisfy the given input and output differences but not satisfy the differential path, such a pair is called *noise* which suggest a set of random keys. It is obvious that if the attacker chooses $\frac{c}{p}$ pairs, he expects to get $\frac{c}{p} \cdot p = c$ right pairs. The value of c depends on the proportion of the probability of the right key being suggested by a right pair to the probability of a random key being suggested by a random pair with the given initial differences. This is called "Signal to Noise ratio" or S/N, which is defined as follows:

$$S/N = \frac{2^k \cdot p}{\gamma \cdot \delta} \tag{1.3}$$

Where k is the number of active bits, p is the probability of differential characteristics, γ is the number of keys suggested by each pair of plaintext and δ is the fraction of the counted pairs among all pairs. It is concluded that if S/N ratio is closed to 1, the number of needed data is high, if it is bigger than 1, the number of data needed is low.

1.3 Linear Cryptanalysis

Linear cryptanalysis, one of the most famous generic statistical attack against block ciphers, was proposed by Matsui and Yamagishi and firstly applied to FEAL block cipher [4]. Then in 1993, Matsui proposed a linear attack on DES cipher [5].

Linear cryptanalysis is a known plaintext-ciphertext attack scenario exploiting linear expressions of some plaintext-ciphertexts and key bits. More explicitly, the attack seeks to have equality between some parity of input, output and key bits. In linear cryptanalysis, the attacker tries to approximate non-linear block cipher using a linear approximation. Let E be an n-bit block cipher, a linear approximation can be defined as the following:

$$\lambda_P \cdot P \oplus \lambda_C \cdot C = \lambda_K \cdot K \tag{1.4}$$

where " $\cdot$ " is the scalar product of two binary strings and λ's are masks which are n-bit binary vectors.

It is expected that for an ideal block cipher, Equation 1.4 holds with probability $\frac{1}{2}$. However,

linear cryptanalysis exploits linear expressions with sufficiently high or low probabilities to distinguish the cipher from a random permutation. The deviation of probability p of occurrences of any linear expression from probability $\frac{1}{2}$ is called *linear probability bias* or *bias*, namely $p - \frac{1}{2}$. The bigger bias is, the better linear cryptanalysis is applicable.

Since the right side of Equation 1.4 is unknown but fixed (it is 0 or 1), there is no mind to write the right side of the equation as 0. if the right side is equal to 1, then only the sign of the bias will change and will not magnitude of bias. For this reason linear approximation can be rewritten as:

$$\lambda_P \cdot P \oplus \lambda_C \cdot C = 0 \tag{1.5}$$

In most block ciphers, the only non-linear components of the round functions are S-boxes. Hence, S-boxes should be approximated linearly by using the Linear Approximation Tables (LATs). Each entry in LATs represent the number of matches between linear expression of input bits and linear expression of output bits. Linear cryptanalysis exploits the entry with highest magnitude.

In linear cryptanalysis, the attacker firstly constructs a linear approximation of the cipher which is equivalent to a differential characteristics, then mount the attack. An $r-1$ rounds linear approximation of the given cipher is constructed by concatenating linear approximations of each round. For example, the approximations of two subsequent rounds are concatenating if the output mask of the first approximation is equal to the input mask of the second approximation. If the first approximation has bias q_1 and the second approximation has bias q_2, then the bias of concatenation of two approximations is $2q_1q_2$ which is calculated according to Matsui's Piling Up Lemma [5]:

Matsui's Piling Up Lemma: Let $X_1, X_2, \ldots X_n$ be independent binary random variables with biases $q_1, q_2, \ldots q_n$, respectively, then

$$Prob(X_1 \oplus X_2 \oplus \ldots \oplus X_n) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} q_i \tag{1.6}$$

or

$$q_{1,2...n} = 2^{n-1} \prod_{i=1}^{n} q_i \tag{1.7}$$

A brief description of r-round linear attack on any block cipher is as follows:

1. Construct an $r-1$ rounds linear approximation with sufficiently high linear probability bias q.
2. Take a set of N plaintext-ciphertexts encrypted under the secret key K.
3. Partially decrypt all ciphertexts through the last round by guessing necessary subkey bits corresponds to masked output bits,
4. For each guessed subkey K_i, count the number of plaintext-ciphertext pairs which satisfy Equation 1.5.

   Let T_i be the number of plaintext-ciphertext pairs satisfy Equation 1.5 for guessed subkey K_i, T_{max} and T_{min} be the maximal and minimal value of $T_i's$, then the actual key bits are determined according to the following algorithm given in [5]:

   - If $|T_{max} - N/2| > |T_{min} - N/2|$, then adopt the key candidate corresponds to T_{max} and guess $\lambda_K \cdot K = 0$ (when $p > 1/2$) or 1 (when $p < 1/2$)
   - If $|T_{max} - N/2| < |T_{min} - N/2|$, then adopt the key candidate corresponds to T_{min} and guess $\lambda_K \cdot K = 1$ (when $p > 1/2$) or 0 (when $p < 1/2$)

The attack is implemented by taking about $\frac{c}{q^{-2}}$ known plaintext-ciphertexts. Note that different values of c chance the success rate of the attack.

1.4 The Structure of Thesis and Our Contributions

In this work, we try to cover cryptanalytic attacks based on related keys. We start giving notion of using related keys, then we continue to give its combinations with differential cryptanalysis. Afterwards, we extend related-key differential cryptanalysis to other combined attacks, namely related-key boomerang, related-key amplified boomerang, related-key rectangle, related-key impossible differential, related-key impossible boomerang and related-key

differential-linear attacks. Finally, we mention slide attack which is not a related-key attack, but its construction is very similar to simple related-key attacks. The aim of this work is to gather all attacks based on related keys and exemplify them.

Our contribution to this work is that we analyse differential property of the key schedule of XTEA (Extended Tiny Encryption Algorithm) and increased the number of rounds of 12-round impossible differential distinguisher for XTEA to 25-round by using related keys with specific differences.

The structure of thesis is as follows. In Chapter 2, basic related-key attacks are mentioned and some examples of these attacks on some block ciphers are described. In Chapter 3, we give the combination of differential cryptanalysis with the related-key idea and some applications of the attack is given. In Chapters 4, 5 and 6, the extensions of related-key differential cryptanalysis, namely related-key boomerang, related-key amplified boomerang, related-key rectangle, related-key impossible differential, related-key impossible boomerang and related-key differential-linear attacks are detailed. In Chapter 7, we give the detailed description of slide attack and its improvements. Firstly, we begin by describing basic related-key attacks.

CHAPTER 2

RELATED-KEY CRYPTANALYSIS

So far, the most well-known attacks on block ciphers are based on the weaknesses of the encryption/decryption algorithm of the ciphers. Therefore, the ciphers are designed to resist the attacks targeting at the cipher itself. However, key scheduling algorithms also play a crucial role in the strength of the cipher. The ciphers which are secure against these attacks may be still insecure, when key scheduling algorithms are taken into consideration. For this reason, by exploiting key scheduling algorithm to get some mathematical relations between keys can enables us to attack the given cipher.

Related-key cryptanalysis is a chosen-key attack, its idea was first introduced by Winternitz and Hellman [6] and improved by Knudsen in 1992 [7] and Biham in 1993 [8], independently. Winternitz and Hellman presented a generic attack which is applicable to all block ciphers. Afterwards, Biham and Knudsen inspired by the work of Winternitz and Hellman, and presented a new chosen-key attack. The first attack was applied to LOKI89 and LOKI91 by Biham [8].

In the related-key attack scenario, the attacker focuses on both structure and key schedule of the cipher to get some weaknesses of the cipher. It is assumed that the attacker only knows the particular relation between the secret keys, but not the actual key values and has a power to make encryptions under the secret key and the derived keys.

Conventional related-key attack introduced by Biham is different from early cryptanalytic methods in a way that it is independent of the number of rounds and mostly round function

of the cipher. Therefore, increasing the number of rounds of the cipher does not help the cipher resist to the conventional related-key attack. Instead of this, key schedule should be redesigned and strengthened.

Related-key attack is adapted to other mostly known powerful attacks and new methods are introduced: related-key differential cryptanalysis, related-key impossible cryptanalysis, related-key boomerang and rectangle attacks are some of them.

Related-key attacks do not seem to be realistic in many real-world cryptographic applications since it is difficult for the attacker to make a sender encrypt plaintexts under related-keys which are unknown to the attacker. However, some real-world cryptographic applications may enable related-key attack. For example, in some applications, encryption program uses secret K, $K + 1$, $K + 2$,... in subsequent encryptions. Moreover, Related-key attacks can be applicable to block cipher based hash functions. More explicitly, if the compression function of an hash algorithm is a block cipher, message words can be seen as related-keys and related-key attacks can be applied. For example, a related-key attack has been applied to AES-192 by Biryukov et al. in 2009 [10]. At first, the attack seems to be unrealistic, however, Biryukov et al. have showed that AES-192 is not an ideal cipher and can not be used as an hash function such as in Davies-Meyer mode.

In this chapter, conventional related-key attacks will be covered. Section 2.1 will include the attack of Winternitz and Hellman which introduced the chosen-key idea [6]. Then, section 2.2 will describe conventional related-key attacks on block ciphers. Finally, Sections 2.3, 2.4 and 2.5 exemplify simple related-key attacks on LOKI89, LOKI91 and SHACAL-1, respectively.

2.1 The Chosen-Key Attack of Winternitz and Hellman

The idea of using more than one key in cryptanalysis of block ciphers was introduced by Winternitz and Hellman [6]. They presented a generic attack which is independent of the encryption and key scheduling algorithms of the cipher. In this attack, any block cipher with n-bit key can be broken by using $O(2^{n/2})$ in computation and memory.

The general overview of the attack is as follows:

1. Firstly, flip the first $\frac{n}{2}$ bits of n-bit secret key K in $2^{n/2}$ possible ways and pick a plaintext P, then compute:
   $E_K(P)$, $E_{K\oplus(00\ldots1)}(P)$, $E_{K\oplus(00\ldots10)}(P)$, , $E_{K\oplus(00\ldots11)}(P)$, $\ldots$, $E_{K\oplus(00\ldots11\ldots11)}(P)$.

2. Afterwards, make $2^{n/2}$ encryptions with a set of key which have all possible values in the last $\frac{n}{2}$-bit positions and compute:
   $E_{(00\ldots0)}(P)$, $E_{(00\ldots10\ldots)}(P)$, $\ldots$, $E_{(11\ldots10\ldots0)}(P)$.

3. Finally, insert two sets of two encryptions in item 1-2 in hash table, check the equality of $E_{K\oplus x}(P)$ and $E_{(y0\ldots0)}(P)$ for some x and y, where x and y are the first and last $\frac{n}{2}$-bit of secret key K. If $E_{K\oplus x}(P) = E_{(y0)}(P)$, then the value of (y, x) is the exact value of K.

   Note that we can have a false alarm that can be eliminated by taking 2 or 3 pairs of plaintexts and ciphertexts.

Now, related-key attacks sticking to Biham's attack in general framework are going to be mentioned in the following section.

2.2 Conventional Related-Key Attacks

2.2.1 Related-Key Known Plaintext Attack

Let E be the block cipher with n-bit block length, r rounds and secret key K. Let K_i be the subkey of round i for $1 \leq i \leq r$ generated by secret key K. Denote by $F(x_i, K_i)$ be the round function of the cipher where x_i is the input to round i. Then, as in Biham's attack [8], assume that the key schedule of the cipher is cyclic, i.e. , if the key schedule of the cipher generates the sequence $(K_1, K_2, \ldots, K_r)$ with K, then it should be possible to generate the sequence $(K_2, K_3, \ldots, K_r, K_1)$ with another key K^*. Under this assumption, we can capture all key bits of K_1. However, due to the strength of the assumption, application of the attack is infeasible, as key schedule of most of the block ciphers do not generate such subkey sequences.

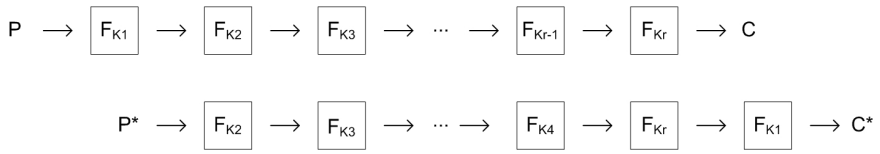

Figure 2.1: Conventional related-key attack

Under this assumption, the main objective of the attack is to identify a pair of plaintexts (P, C) and (P^*, C^*) that satisfy the equation $F(P, K_1) = P^*$ where C and C^* are corresponding ciphertexts of P and P^* under K and K^*, respectively. By this way, two encryption processes will follow the same way in $r - 1$ rounds as seen in Figure 2.1 and the relation between ciphertexts $F(C, K_1) = C^*$ is obtained for free. Such a pair is called *a slid pair* and enables us to identify the subkey K_1 by using the equations $F(P, K_1) = P^*$ and $F(C, K_1) = C^*$.

The general case of attack on any block ciphers is as follows: Take two sets of $2^{n/2}$ plaintexts P_i and P_j^* and obtain their corresponding ciphertexts C_i and C_j^* under K and K^*, respectively. By the birthday paradox, it is expected to get only one slid pair. Then, for each pair (P_i, C_i) and (P_i^*, C_i^*) solve the equations $F(P_i, K_1) = P_j^*$ and $F(C_i, K_1) = C_j^*$ until to find K_1 which satisfies both equations. Due to the structure of the cipher, the equations $F(P_i, K_1) = P_j^*$ and $F(C_i, K_1) = C_j^*$ are not both satisfied if P_i and P_j^* is not a slid pair, thus, we probably deal with a slid pair. Once a slid pair is identified, then conclude that K_1 which verifies the system is equal to the actual value of K_1. Notice that, another assumption of conventional related-key attack is that the round function F must be a *weak* permutation to let us derive K_1, that is with a few slid pairs it is possible to identify K_1 given $F(P_i, K_1) = P_j^*$ and $F(C_i, K_1) = C_j^*$.

The cipher is broken by taking $O(2^{n/2})$ related-key known plaintext-ciphertexts (P_i, C_i) and (P_j^*, C_j^*) with time complexity $O(2^n)$, since we have $O(2^n)$ plaintext-ciphertext pairs to examine.

2.2.2 Conventional Related-Key Attacks on Ciphers with Feistel Structure:

Related-key attack can be applied to Feistel ciphers with lower data and time complexities due to the structure of the Feistel ciphers. In the case of generic Feistel ciphers, a pair (P, C) and (P^*, C^*) forms a slid pair if the relation between plaintexts $F((P_L, P_R), K_1) = (P_R, P_L \oplus f(P_R \oplus$

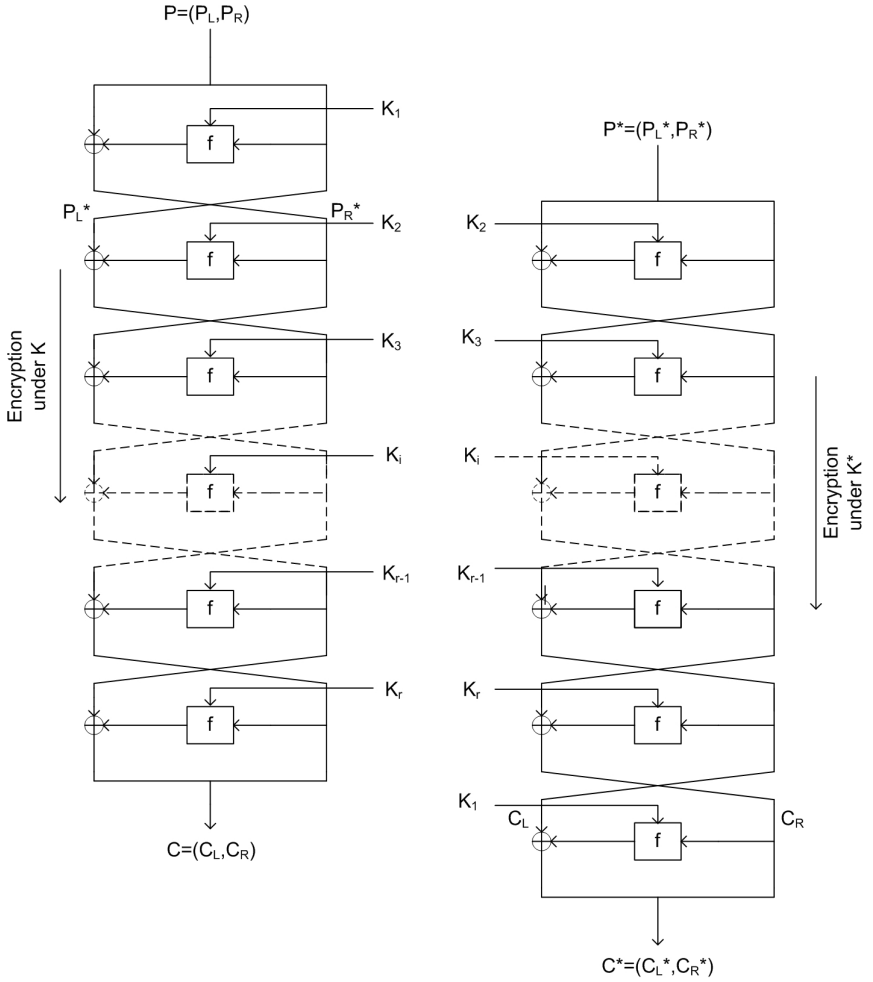

Figure 2.2: Conventional related-key attack on a generic Feistel cipher

$K_1)) = (P_L^*, P_R^*) = P^*$ holds which makes the relation between the ciphertexts $F(C_R, C_L) = (C_R, C_L \oplus f(C_R \oplus K_1)) = (C_L^*, C_R^*)$ hold for free. Then, the relations of a slid pair for generic Feistel ciphers are:

$$P_L^* = P_R, \tag{2.1}$$

$$P_R^* = P_L \oplus F(P_R, K_1), \tag{2.2}$$

$$C_R^* = C_L, \tag{2.3}$$

$$C_L^* = C_R \oplus F(C_R^*, K_1). \tag{2.4}$$

Mainly, there are two types of related-key attack on Feistel ciphers: a related-key known plaintext-ciphertext attack and a related-key chosen plaintext-ciphertext attack.

- *A related-key known plaintext-ciphertext attack*:

  Take two sets of $2^{n/2}$ plaintexts $P_i's$ and $P_j^{*}'s$ and obtain their corresponding ciphertexts $C_i's$ and $C_j^{*}'s$ under K and K^*, respectively. By the birthday paradox, it is expected to find only one slid pair. Then, Equation 2.1 forces $\frac{n}{2}$-bit filtering condition on plaintext pairs. On the other hand, by Equation 2.3, there is also $\frac{n}{2}$-bit filtering condition on ciphertext pairs. In total, there are n-bit filtering condition on a slid pair which eliminates wrong slid pairs. This elimination can be done by sorting all plaintext-ciphertext pairs into a hash table and checking Equations 2.1 and 2.3. Then, we expect to get a false alarm and a slid pair after filtering. For two pairs, Equations 2.2 and 2.4 to derive the correct value of K_1. In addition, it is easy to eliminate the key suggested by the false alarm by just trying a few plaintext-ciphertext pairs once K_1 have identified.

  This attack has $O(2^{n/2})$ data complexity, $O(2^{n/2})$ work including filtering by using hash tables.

- *A related-key chosen plaintext-ciphertext attack*:

  In the case of Feistel ciphers, the data complexity of the attack can be reduced from $O(2^{n/2})$ to $O(2^{n/4})$ by taking carefully chosen plaintexts satisfying Equation 2.1. More

explicitly, the attacker picks two sets of $2^{n/4}$ plaintext-ciphertexts such that one set is of the form (P_L^i, A) and another set is of the form (A, P_R^j), respectively where A is $\frac{n}{2}$-bit fixed value. These sets constitute $2^{n/2} \cdot 2^{n/2} = 2^{n/2}$ plaintext-ciphertext pairs one out of which is a slid pair by the birthday paradox. On the other hand, by Equation 2.3, there is $\frac{n}{2}$-bit filtering condition on ciphertexts that eliminates all wrong pairs. Hence, we have one slid pairs and a false alarm after filtering. Once a slid pair have found, we can extract the correct key value of K_1 from Equations 2.2 and 2.4. Finally, this attack has $O(2^{n/4})$ data complexity and $O(2^{n/4})$ offline work.

2.3 A Related-Key Attack on LOKI89

In this subsection, a related-key attack of Biham against LOKI89 based on two related keys is presented [8]. This attack is a key recovery attack that enables to identify all bits of the secret key K.

Firstly, notations used in the attack and description of LOKI89 are given:

(K_L, K_R): 32-bit left and right halves of 64-bit secret key K,
$<<< i$: A cyclic shift to the left by i bits.

A Brief Description of LOKI89 and LOKI91:

LOKI89 and LOKI91 are a family of block ciphers designed by Brown et al. [9] for the replacement of DES. Therefore, they have very similar structure with DES. LOKI89 and LOKI91 are 16 rounds Feistel-type block ciphers with 64-bit block and key sizes. Like DES, they have permutation P, expansion E and S-boxes in its f-function. Different from DES, LOKI89 has initial and final key whitenings, while LOKI91 does not have. Notice that LOKI91 is an improved version of LOKI89 in a way that it has stronger S-box and key schedule than LOKI89 has.

The Key Schedule of LOKI89:

The crucial part of LOKI89 is its cyclic key schedule which makes the cipher vulnerable to related-key attacks. The key schedule of LOKI89 has Feistel structure that it uses only cyclic shifts to generate subkeys. More explicitly, 64-bit secret key is split into equal halves K_L and K_R. Then, replace K_L by K_1 and K_R by K_2 and other subkeys K_i of round i are obtained by rotating K_{i-2} by 12 bits to the left, i.e $K_i = (K_{i-2} <<< 12)$.

The Key Schedule of LOKI91:

Unlike LOKI89, LOKI91 does not have a cyclic key schedule, however, it is still nonresistant against related-key attacks. The key schedule of LOKI91 has a Feistel structure which splits 64-bit secret key K into equal halves K_L and K_R. Then, replace K_L by K_1 and $(K_L <<< 12)$ by K_2 and other subkeys of round i are obtained by rotating the subkeys K_{i-4} by 25 bits to the left ($K_i = K_{i-4} <<< 25$).

The structure of LOKI89 can be seen in Figure 2.3.

The attack

First of all, the attack starts by choosing the relation between two related-keys K and K^* such that $K^* = (K_R, (K_L <<< 12))$, then we have the relation between the subkeys as:

$$\begin{aligned} K_2 &= K_1^* \\ K_3 &= K_4^* \\ &\vdots \\ K_{16} &= K_{15}^* \\ K_1 &= K_{16}^* \end{aligned} \tag{2.5}$$

Emphasize that Equality 2.5 holds due to the cyclic property of the key schedule of LOKI89. Now, the cricual assumption is that input to the second round in the encryption under key K is equal to the input to the first round in the encryption under key K^*, namely $F(P \oplus K, K_1) = P^* \oplus K^*$, since $F(P \oplus K, K_1) = (P_R \oplus K_R, P_L \oplus K_L \oplus f(P_R \oplus K_R, K_L))$ and $P \oplus K^* =$

$(P_L^* \oplus K_R, P_R^* \oplus (K_L <<< 12))$, we have

$$P^* = (P_L^*, P_R^*) = (P_R, P_L \oplus K_L \oplus (K_L <<< 12) \oplus f(P_R \oplus K_R, K_L)) \tag{2.6}$$

Then, two encryption processes follow the same way in the subsequent 15 rounds and at the end the following relation between ciphertexts are obtained for free:

$$C^* = (C_L^*, C_R^*) = (C_R \oplus K_L \oplus (K_L <<< 12) \oplus f(C_R \oplus K_R, K_L), C_L) \tag{2.7}$$

Then, by using Equations 2.6 and 2.7, the attack can be performed as follows:

1. Take two sets of 2^{16} plaintext-ciphertext pairs satisfying $P_L^* = P_R$: one set of plaintexts is of the form (P_L^i, A) and another one is of the form (A, P_R^j), respectively where A is 32-bit fixed value.
   The number of pairs is $2^{16} \cdot 2^{16} = 2^{32}$ and by the Birthday Paradox, one slid pair exists.

2. Ask for encryption of plaintexts $P_i's$ and $P_j^{*\prime}s$ under K and K^* respectively, and obtain their corresponding ciphertexts $C_i's$ and $C_j^{*\prime}s$.

3. Due to Equation 2.7, there are 32-bit filtering condition on ciphertext pairs. Thus, after filtering one slid pair and one false alarm left.

4. Combining Equations 2.6 and 2.7, we have

$$f(P_R \oplus K_R, K_L) \oplus f(C_L \oplus K_R, K_L) = P_R^* \oplus P_L \oplus C_L^* \oplus C_R \tag{2.8}$$

In Equation 2.8, P_R, P_L, P_R^*, C_L, C_R and C_L^* are known values and only unknown value is $K_R \oplus K_L$. By using Difference Distribution Table of S-boxes of LOKI89, $K_L \oplus K_R$ can be identified easily. Once the value of $K_L \oplus K_R$ have been found, then by using Equation 2.7, K_L and K_R can be found immediately.

2.3.1 A Related-Key Attack on Block Ciphers with More Generalized Key Schedule

The conventional related-key attack mentioned in the previous section can be generalized in a way that the key schedule of the cipher generates the sequence $(K_1, K_2, \ldots, K_r)$ with K and can generate another the sequence $(K_2, K_3, \ldots, K_r, K_{r+1})$ with another key K^*. This assumption

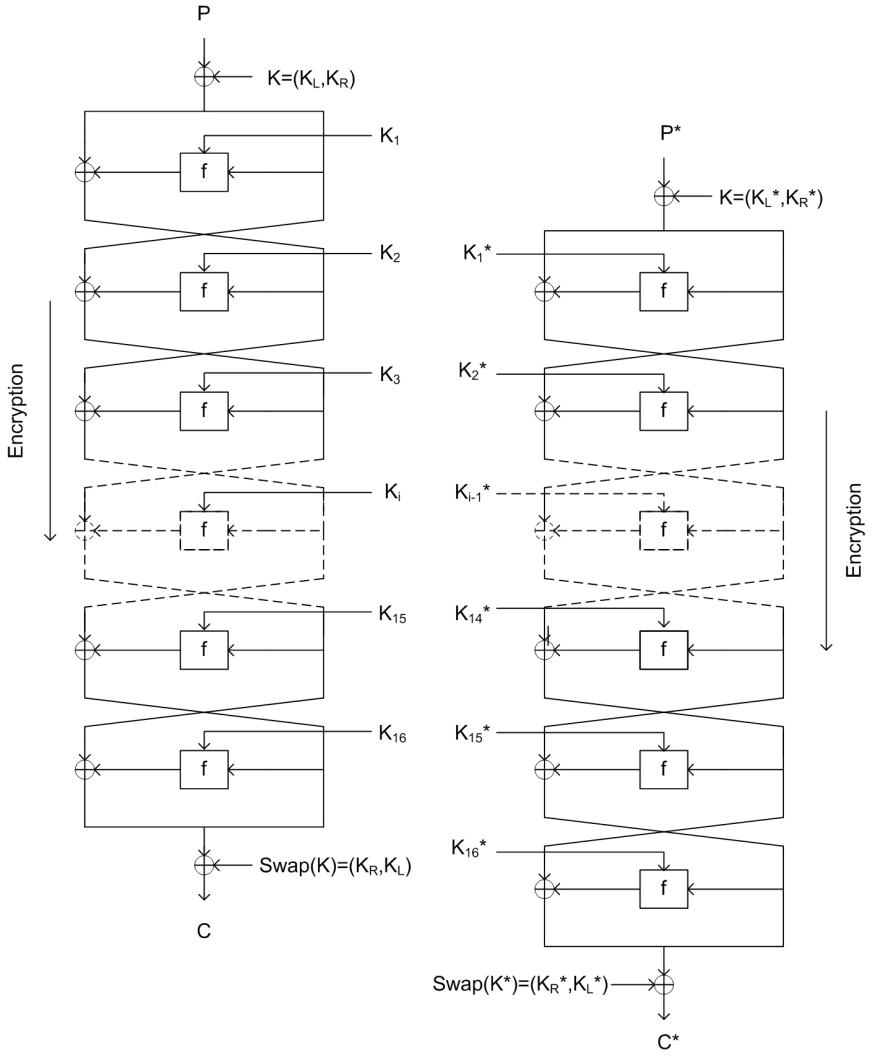

Figure 2.3: Related-key attack on full-round LOKI89

is weaker than the previous one, since the key schedule need not to be cyclic. However, the attack is not possible to mount, because, a pair (P, C) and (P^*, C^*) forms a slid pair if they satisfy the relations $F(P, K_1) = C$ and $F(P^*, K_{r+1}) = C^*$. Since identifying a slid pair is not trivial and it is not possible to solve these two equations if K_1 and K_{r+1} do not have almost all their bits in common. But, this situation can be handled only if there are some filtering conditions on candidates slid pairs. Thus, it is concluded that this attack can be feasible if it is applied to ciphers with Feistel structure. The attack procedure is as follows: we choose one set of $2^{n/2}$ plaintext-ciphertexts (P, C) encrypted with K and another set of plaintext-ciphertexts (P^*, C^*) encrypted with K^*, then there are 2^n pairs (P, C) and (P^*, C^*) in total. Due to the design of Feistel ciphers, there are two $\frac{n}{2} - bit$ filtering conditions on plaintext and ciphertext pairs, namely $P_L^* = P_R$ and $C_L^* = C_R$. Thus, after filtering one slid pair remains and suggested pairs of (K_1, K_{r+1}) are extracted. But, there are many suggested pairs of (K_1, K_{r+1}). If K_1 and K_{r+1} have common bits, then the attack can be succeeded due to the filtering of common bits on suggested key pairs.

2.4 A Related-Key Attack on full round LOKI91

LOKI91 has a different key schedule from LOKI89 in a way that it does not allow to make related-key attack by shifting one round of encryption process, because, its subsequent subkeys are generated by using different number of shifts and is independent of the subkeys of rounds i and $i - 1$. Therefore, an encryption process is shifted by two rounds against another one and a related-key attack to LOKI91 is applied.

Now, let $K = (K_L, K_R)$ be the secret key and $K^* = (K_L^*, K_R^*)$ the related-key. If an encryption under K^* is shifted by two rounds against an encryption under K and the output of the second round of an encryption under K is equal to the plaintext of an encryption under K^*, then two encryption process will follow the same way in subsequent 12 rounds. More explicitly, the relation between plaintexts will be:

$$P^* = (P_L^*, P_R^*) = (P_L \oplus f(P_R \oplus K_1), P_R \oplus f(P_L \oplus f(P_R \oplus K_1) \oplus K_2)) \tag{2.9}$$

Then, a similar relation between ciphertexts will be obtained for free:

$$C^* = (C_L^*, C_R^*) = (C_R \oplus f(f(C_R \oplus K_{15}^*) \oplus C_L \oplus K_{16}^*), f(C_R \oplus K_{15}^*) \oplus C_L) \tag{2.10}$$

Finding a slid pair satisfying Equations 2.9 and 2.10 is not evident, because K_1 and K_2 are unknown. However, since $K_1 = (K_1 <<< 12)$, that is K_1 and K_2 shares all their bits, and that makes finding slid pair easier.

A brief description of related-key chosen plaintext attack on LOKI91 is given as follows:

- Choose a set of 2^{16} plaintexts $P_i's$ and obtain their corresponding ciphertexts $C_i^{'s}$.
- Guess 2^{32}-bit values of K_1 and K_2 and by using Equation 2.9 calculate the output data of second round of encryption under K which are assumed to be equal to plaintexts $P_j's$ of an encryption under K^*. Therefore, we have a set of $2^{16} \cdot 2^{32} = 2^{48}$ $P_j's$ in total.
- Ask for encryption of $P_j^{*'}s$ under K^* and obtain their corresponding ciphertexts $C_j^{*'}s$. For each pair $((P_i, C_i), (P_j^*, C_j^*))$ check if they satisfy Equation 2.10 with K_{15}^* and K_{16}^* or not. Then, keep only the pairs that satisfy Equation 2.10. The remaining pair is a slid pair with a high probability since, Equations 2.9 and 2.10 cannot both hold if $((P_i, C_i), (P_j^*, C_j^*)$ do not form a slid pair. Once a slid pair is identified, K_1 which is the left part of secret key K can be immediately found.

Next, a full-round simple related-key attack on SHACAL-1 presented by Biham et al. will be covered [11].

2.5 Related-Key Cryptanalysis of full-round SHACAL-1

A simple related-key attack on full round SHACAL-1 was presented by Biham et al. [11]. The attack captures 548 bits of the secret key K. SHACAL-1 is a 80-round Feistel-type block cipher proposed by Handschuh and Naccache to the NESSIE project in 2000 [12]. It has 160-bit block size and variable key sizes (0-512). For the detailed description of SHACAL-1, refer to Chapter 5.

This attack is applicable to SHACAL-1, because SHACAL-1 has linear key schedule. The attack starts by choosing relations between the secret key and the related-key:

Let $K = (K_0, K_1, \ldots, K_{15})$ be the secret key and $K^* = (K_0^*, K_1^* \ldots, K_{15}^*)$ be the related-key. Then, K and K^* satisfy $K_i^* = K_{i+1}$ for $0 \leq i \leq 78$, more explicitly:

$$K_i^* = \begin{cases} K_{i+1}, & 0 \leq i \leq 14 \\ K_{16} = (K_1 3 \oplus K_8 \oplus K_2 \oplus K_0) & i = 15 \end{cases}$$

If it is assumed that the output of the first round of an encryption under K is equal to the input to the first (plaintext) round of an encryption under K^*, then the subsequent 19 rounds in both encryption processes will be the same. However, in round 20 of an encryption under K uses different f_i from an encryption under K^* in round 19, thus, we need to keep sliding probabilistically; the input to the round 21 in the first encryption under K should be equal to the input to the round 22 in an encryption under K^*, that is:

$$A_{21} = A_{20}^*, B_{21} = B_{20}^*, C_{21} = C_{20}^*, D_{21} = D_{20}^* \text{ and } E_{21} = E_{20}^*.$$

Due to the Feistel structure of SHACAL-1, we only need the equality $A_{21} = A_{20}^*$, that is:

$$\begin{aligned} A_{21} &= K_{20} + (A_{20} <<< 5) + f_{20}(B_{20}, C_{20}, D_{20}) + E_{20} + \delta_{20} \\ &= A_{20}^* + (A_{19}^* <<< 5) + f_{19}(B_{19}^*, C_{19}^*, D_{19}^*) + E_{19}^* + \delta_{19} \end{aligned}$$

The equality of two 32-bit blocks holds with probability of 2^{-32} for a random permutation. Hence, for SHACAL-1, the probability is 2^{-32} which is experimentally determined in [11]. Then, sliding continues until round 40 of the encryption under K. Therefore, sliding will be kept by the assumption of the equality in round 41 and round 40 in both encryptions. But, the equality happens with a probability of 2^{-32}. Afterwards, we come up against same situation in round 60. Thus, we make same assumption as before and the equality holds with probability of 2^{-32}.

Hence, we can conclude that the probability of a pair form a slid pair with satisfying Equation 2.11 is $(2^{-32})^3 = 2^{-96}$.

2.5.1 An Attack on full-round SHACAL-1

The attack algorithm can be described as follows:

1. Pick 2^{67} pairs of structures of 2^{32} chosen plaintexts such that the set $\mathcal{P}$ contains plaintexts (A, B, C, D, x) under K where A, B, C, D and E are fixed and for all possible values of x and the set $\mathcal{P}^*$ contains plaintexts $(y, A, (B <<< 30), C, D)$ under K^* for all possible values of y. Obtain the set of their corresponding ciphertexts C and C^*, respectively.

   Each structure constitutes $2^{32} \cdot 2^{32} = 2^{64}$ chosen plaintexts, therefore, in total there are $2^{67} \cdot 2^{64} = 2^{131}$ pairs. Note that for each $P \in \mathcal{P}$, there is P^* such that $P^* \in \mathcal{P}^*$, therefore, with 2^{67} pairs of structures of 2^{32}, it is expected to have $2^{67} \cdot 2^{32} \cdot 2^{-96} = 2^3$ slid pairs.

2. Since $A_{80} = A^*_{79}$, $B_{80} = B^*_{79}$, $C_{80} = C^*_{79}$, $D_{80} = D^*_{79}$ and $E_{80} = E^*_{79}$, we have 128-bit filtering condition on ciphertexts. Therefore, keep only the ciphertext pairs (C, C^*) such that $C = (a, b, c, d, e)$ and $C^* = (x, a, (b <<< 30), c, d)$. Since there are 2^{131} pairs and due to 128-bit filtering condition, $2^{131} \cdot 2^{-128} = 2^3$ candidate slid pairs remain.

3. For each candidate slid pairs, identify K_0 and K_{80} by the following equations:

$$\begin{aligned} K_0 &= y - ((A <<< 5) + f_{if}(B, C, D) + x + \delta_0)), \\ K_{80} &= a^* - ((a <<< 5) + f_{xor}(b, c, d) + e + \delta_{79})). \end{aligned}$$

4. For each candidate slid pairs, output (K_0, K_{80}).

The attack can be applied once more by pairing keys as (K^*, K^{**}), (K^{**}, K^{***}),..., (K^{6*}, K^{7*}). For example,

$$K^{**}_i = \begin{cases} K^*_{i+1}, & 0 \le i \le 14 \\ K^*_{16} = (K^*_{13} \oplus K^*_8 \oplus K^*_2 \oplus K^*_0) & i = 15 \end{cases}$$

If steps 1-3 are repeated for keys K^* and K^{**}, all bits of subkeys K^*_0 and K^*_{80} which are equivalent to K_1 and K_{81} are obtained. Repeating the attack 7 times, $64 \cdot 7 = 448$ linear equations of subkeys are obtained due to the linearity of the key schedule of SHACAL-1 which enables to identify 448 bits of the secret key. The remaining 64-bit of the secret key can be found by exhaustive search.

CHAPTER 3

RELATED-KEY DIFFERENTIAL CRYPTANALYSIS

The related-key scenario was combined with differential cryptanalysis by Kelsey et al. in 1996 [13] and was firstly applied to IDEA, G-DES, GOST, SAFER and Triple-DES.

Related-key differential cryptanalysis both exploits differential property of structure and key schedule algorithm of the cipher. Namely, the attacker can choose specific differences between plaintexts and keys. The attack procedure is same as the attack procedure of differential attack mentioned in Section 1.2 except the attacker uses more than one keys to make encryptions.

The overview of this chapter is as follows: In section 3.1, a related-differential cryptanalysis of full-round GOST proposed by Ko et al. [14] is given. Then, in Section 3.2, a related-key attack on 5-round and 6-round KASUMI proposed by Blunden et al. in 2001 [15] is described.

3.1 Related-Key Differential Cryptanalysis of GOST

The GOST block cipher, its name is an abbreviation for "Gosudarstvennyi Standard " or "Government Standard", was developed by the former Soviet Union in 1989 [16] and became as a standard for Russian Federation. Since GOST was developed to be an alternative to DES, the encryption algorithm of GOST is very similar to that of DES. However, unlike DES, some components of the encryption algorithm of the cipher (S-boxes) are previously kept secret, then it became public.

GOST is vulnerable to differential attacks, because the encryption algorithm does not satisfy diffusion well. Besides, key scheduling algorithm of GOST does not satisfy difussion well, either, due to its simple structure. Therefore, by exploiting these weaknesses the structure and the key schedule of the cipher, several differential attacks are applied to the cipher [17, 14].

In this section, a related-key differential attack on full round GOST which was proposed by Ko et al. [14] is given.

3.1.1 Notations

The notations used in the description of GOST and in the attack are given as follows:

$\oplus$: Exclusive-OR operation
$\boxplus$: Addition in modulo 2^{32}
($<<< i$): i-bit cyclic rotation to the left
K: A 256-bit master key
K_i: ith 32-bit of master key K, $1 \leq i \leq 8$
$(X_{i,L}, X_{i,R})$: Left and right halves of input blocks to the round i, respectively, $1 \leq i \leq 64$
e_i: A 32-bit block having zeros in all positions except the position i
?: Any difference.

3.1.2 The GOST Block Cipher

GOST is a 64-bit block cipher with 256-bit secret key and has a 32-round iterative Feistel structure. It has simple f-function which consists of a subkey addition in modulo 2^{32}, eight different 4×4 S-boxes $S_1, S_2, \ldots S_8$ and 11-bit cyclic rotation to the left.

3.1.2.1 Key Scheduling Algorithm

GOST has a quite simple key scheduling algorithm such that each round subkey is a part of the secret key K. More specifically, A 256-bit secret key K is divided into eight 32-bit parts $(K_1, K_2, ..., K_8)$ to generate 32-bit subkeys k_i, $1 \leq i \leq 32$ and then each subkey is one of the 8

parts of K. The subkeys used in each round are specified in Table 3.1.

Table 3.1: The Key Schedule of GOST

Round	Subkey	Round	Subkey	Round	Subkey	Round	Subkey
1	K_1	9	K_1	17	K_1	25	K_8
2	K_2	10	K_2	18	K_2	26	K_7
3	K_3	11	K_3	19	K_3	27	K_6
4	K_4	12	K_4	20	K_4	28	K_5
5	K_5	13	K_5	21	K_5	29	K_4
6	K_6	14	K_6	22	K_6	30	K_3
7	K_7	15	K_7	23	K_7	31	K_2
8	K_8	16	K_8	24	K_8	32	K_1

3.1.2.2 Encryption Algorithm

Encryption function of GOST is an 32 times iteration of the round function depicted in Figure 3.1 which splits 64-bit input into two 32-bit halves and then applies f-function to one half of its input. The f-function used in each round can be defined as follows:

$f : \{0,1\}^{32} \times \{0,1\}^{32} \rightarrow \{0,1\}^{32}$, $f(x,k) = (S_8((x \boxplus k)_8) \| S_7((x \boxplus k)_7) \| \ldots \| S_1((x \boxplus k)_1)) <<< 11$, where x is any 32-bit value, k is 32-bit subkey and $x \boxplus k = ((x \boxplus k)_8, (x \boxplus k)_7, \ldots, (x \boxplus k)_1)$

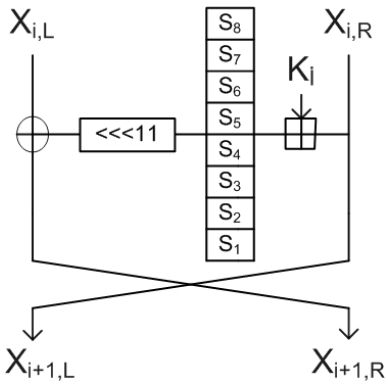

Figure 3.1: ith round function of GOST

3.1.3 A Distinguishing Attack on Full Round GOST

Ko et al. [14] presented a distinguishing attack on full round GOST by building a 32-round related-key differential with probability 1. This 32-round differential characteristic depicted in Table 3.1.3 is constructed by choosing the difference between two plaintext as $P \oplus P^* = (P_L \oplus P_L^*, P_R \oplus P_R^*) = (e_{31}, e_{31})$ and the difference between two related-keys as $K \oplus K^* = (K_1 \oplus K_1^*, K_2 \oplus K_2^*, \ldots, K_8 \oplus K_8^*) = (e_{31}, e_{31}, \ldots, e_{31})$. It is seen that the input difference to each round is always (e_{31}, e_{31}) because plaintext difference is preserved due to the zero input difference to S-boxes in each round with probability 1.

Since the probability of the related-key differential distinguisher is 1, the probability of distinguishing the GOST cipher from a truly random permutation is $(1 - 2^{-64})$. Therefore, the attack can be implemented by choosing only two plaintext-ciphertext pairs with the given input differences.

Table 3.2: A 32-Round Related-Key Differential Characteristic for GOST

	$\Delta X_{i,L}$	$\Delta X_{i,R}$	ΔK_i
ΔX_1	e_{31}	e_{31}	e_{31}
ΔX_2	e_{31}	e_{31}	e_{31}
ΔX_3	e_{31}	e_{31}	e_{31}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
ΔX_{31}	e_{31}	e_{31}	e_{31}
ΔX_{32}	e_{31}	e_{31}	e_{31}
Output	e_{31}	e_{31}	-

3.1.4 A Related-Key Differential Attack on Full Round GOST

This attack utilizes 30-round related-key differential characteristics with probability of 2^{-30}. The attack recovers 4 bits of the last round subkey K_1, however in the original paper there exist extra work to find other 8 bits of K_1.

3.1.4.1 A 30-Round Related-Key Differential Characteristics for GOST

If the difference between two plaintexts $P = (P_L, P_R)$ and $P^* = (P_L^*, P_R^*)$ are chosen as $P \oplus P^* = (P_L \oplus P_L^*, P_R \oplus P_R^*) = (e_{30}, e_{30})$ and the difference between two keys K and K^* are chosen as

$K \oplus K^* = (K_1 \oplus K_1^*, K_2 \oplus K_2^*, \ldots, K_8 \oplus K_8^*) = (e_{30}, e_{30}, \ldots, e_{30})$ then the output difference of round 30 will be (e_{30}, e_{30}) with probability of 2^{-30}. More explicitly, if the input difference to the round is (e_{30}, e_{30}) and subkey difference is (e_{30}), after the key addition, the input difference to the s-boxes will be zero with a probability of 2^{-1}. By this way, one can build a 30-round related-key differential characteristics with probability of 2^{-30} as depicted in Table 3.1.4.1.

Table 3.3: A 30-Round Related-Key Differential Characteristic for GOST

	$\Delta X_{i-1,L}$	$\Delta X_{i-1,R}$	ΔK_i	Probability
ΔX_1	e_{30}	e_{30}	e_{30}	2^{-1}
ΔX_2	e_{30}	e_{30}	e_{30}	2^{-1}
ΔX_3	e_{30}	e_{30}	e_{30}	2^{-1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
ΔX_{31}	e_{30}	e_{30}	e_{30}	2^{-1}
ΔX_{32}	e_{30}	e_{30}	e_{30}	2^{-1}
Output	e_{30}	e_{30}	-	-

3.1.4.2 A Full Round Attack on GOST

The attack starts by imposing extra conditions on ciphertext pairs. Therefore, it is assumed that ciphertext differences are $C_R \oplus C_R^* = e_{30}$ and $C_L \oplus C_L^*[0-6] = 0$, $C_L \oplus C_L^*[7-10] =?$, $C_L \oplus C_L^*[11-29] = 0$, $C_L \oplus C_L^*[30] = 1$ and $C_L \oplus C_L^*[31] = 0$. Then, this condition needs input difference to S_8 in round 31 is zero and the input difference to S_8 in round 32 is nonzero.

The attack algorithm can be explained as follows:

- Choose 2^{35} plaintext pairs P_i and P_i^* such that $P_i \oplus P_i^* = (e_{30}, e_{30})$ and obtain their corresponding ciphertexts C_i and C_i^* under K and K^*, respectively.
- Keep only the ciphertext pairs C_i and C_i^* such that $C_R \oplus C_R^* = e_{30}$ and $C_L \oplus C_L^*[0-6] = 0$, $C_L \oplus C_L^*[7-10] =?$, $C_L \oplus C_L^*[11-29] = 0$, $C_L \oplus C_L^*[30] = 1$ and $C_L \oplus C_L^*[31] = 0$.
- For each 4-bit key candidate, check two equations given below for each remaning ciphertext pairs and count the number of ,

  $S_8(C_R[28-31] + K_1[28-31]) \oplus C_L[7-10] = S_8(C_R^*[28-31] + K_1^*[28-31]) \oplus C_L^*[7-10]$

  or

  $S_8(C_R[28-31] + 1 + K_1[28-31]) \oplus C_L[7-10] = S_8(C_R^*[28-31] + 1 + K_1^*[28-31]) \oplus C_L^*[7-10]$

If a key candidate is counted at least four times, then conclude that this key candidate is the actual key.

3.2 Related-Key Differential Cryptanalysis of KASUMI

Third Generation Partnership Project (3GPP) aims to standardize the future mobile telephony. The goal of 3GPP is to provide more secure environment then GSM. Two main objectives of the design of 3GPP confidentially and integrity algorithms are to provide high security and low cost implementation in hardware. KASUMI [18] which a basis for a confidentiality algorithm f_8 and integrity algorithm f_9 was adapted from the block cipher MISTY1 [19] by strengthening the algorithm of MISTY1 and simplifying the implementation without compromising the complexity and security, respectively.

In 2005, Biham et al. [20] proved that KASUMI is not secure against differential based related-key attacks. They proposed a related-key rectangle attack on full round KASUMI which is faster than exhaustive search. In this section, we will describe the related-key differential attack of Blunden et al. [15] that was proposed in 2001. This is not the best attack on KASUMI, however, it is important that it constitutes a basis for the full round attack on KASUMI.

3.2.1 Notations

The notations used in the specification of KASUMI and description of the attack are listed as follows:

|| : The concatenation of two string
<<< i: i-bit rotation to the left
$\vee$: The bitwise OR operator
$\wedge$: The bitwise AND operator
a^n: An n-bit string with full of a where $a \in \{0, 1\}$

3.2.2 The Block Cipher KASUMI

KASUMI is an 8-round Feistel block cipher which encrypts an 64-bit input block into 64-bit output block by using a 128-bit user key. It uses a linear key scheduling algorithm which generates 32 subkeys of 16-bit length by using 128-bit secret key K. The key schedule algorithm and the components of KASUMI are described in the following subsection.

3.2.2.1 The Key Scheduling Algorithm of KASUMI

KASUMI uses very simple linear key scheduling algorithm that makes the cipher vulnerable to related-key attacks. The subkeys KL_i, KO_i and KI_i specified in Table 3.4 are derived from the secret key K in the following way:

The 128-bit user key K is partitioned into eight 16-bit sections $K_1, K_2 \ldots K_8$ where $K = K_1 \| K_2 \| \ldots K_8$. Then, each section is used exactly once in the generation of the subkeys of one round in a way that either it is rotated to the left or it is added with a constant. More explicitly, K_i' is an 16-bit subkey value that is derived from K_i such that $K_i' = K_i \oplus \delta_i$, for each i, $1 \leq i \leq 8$. Here, δ_i is an 16-bit constant value which is given in Table 3.5.

Table 3.4: The Key Schedule of KASUMI

Round i	$KL_{i,1}$	$KL_{i,2}$	$KO_{i,1}$	$KO_{i,2}$	$KO_{i,3}$	$KI_{i,1}$	$KI_{i,2}$	$KI_{i,3}$
1	K_1 <<< 1	K_3'	K_2 <<< 5	K_6 <<< 8	K_7 <<< 13	K_5'	K_4'	K_8'
2	K_2 <<< 1	K_4'	K_3 <<< 5	K_7 <<< 8	K_8 <<< 13	K_6'	K_5'	K_1'
3	K_3 <<< 1	K_5'	K_4 <<< 5	K_8 <<< 8	K_1 <<< 13	K_7'	K_6'	K_2'
4	K_4 <<< 1	K_6'	K_5 <<< 5	K_1 <<< 8	K_2 <<< 13	K_8'	K_7'	K_3'
5	K_5 <<< 1	K_7'	K_6 <<< 5	K_2 <<< 8	K_3 <<< 13	K_1'	K_8'	K_4'
6	K_6 <<< 1	K_8'	K_7 <<< 5	K_3 <<< 8	K_4 <<< 13	K_2'	K_1'	K_5'
7	K_7 <<< 1	K_1'	K_8 <<< 5	K_4 <<< 8	K_5 <<< 13	K_3'	K_2'	K_6'
8	K_8 <<< 1	K_2'	K_1 <<< 5	K_5 <<< 8	K_6 <<< 13	K_4'	K_3'	K_7'

Table 3.5: Constants Used in the Key Schedule of KASUMI

δ_1	δ_2	δ_3	δ_4	δ_5	δ_6	δ_7	δ_8
0x0123	0x4567	0x89AB	0XCDEF	0XFEDC	0XBA98	0X7654	0X3210

3.2.2.2 The Encryption Algorithm of KASUMI

Each round of encryption function of KASUMI has two S-boxes S_7 and S_9 and three functions the FO function, the FL function and the FI function.

The FO function is a 3-round balanced Feistel construction which mixes 32-bit data with two 48-bit subkeys in each round. Each round of FO function includes key addition step and the FI function. The FI function is also a 4-round unbalanced Feistel construction which takes 16-bit input data and 16-bit subkey. Each round has an S-box such that even rounds use S_7 and and odd rounds use S_9. The FL function is a 2-round Feistel construction which takes 32-bit input data and 32-bit subkey. It has two nonlinear operations; the OR operation and the AND operation both of which operates on 16-bit data and 16-bit subkey. S-boxes used in the encryption function are S_7 and S_9, S_7 which maps 7-bit input into 7-bit output, and S_9 which maps 9-bit input into 9-bit output.

The encryption function of KASUMI has three f-functions and two S-boxes described as follows:

The FL Function: The FL function accepts 32-bit input value X_0 and 32-bit subkey KL_i and separate the input X_0 into two parts $X_{0,L}$ and $X_{0,R}$ such that $X_0 = X_{0,L}||X_{0,R}$. Then, it applies the following algorithm:

$$X_{1,R} = X_{0,R} \oplus ((X_{0,L} \wedge KL_{i,1}) <<< 1)$$
$$X_{1,L} = X_{0,L} \oplus ((X_{1,R} \vee KL_{i,2}) <<< 1)$$

where $KL_i = KL_{i,1}||KL_{i,2}$, and $X_1 = X_{1,L}||X_{1,R}$ is the output of the function FL in round i.

The FI Function: The function FI takes a 16-bit X_0 and a 16-bit subkey $KI_{i,j}$ as an input, then it divides 16-bit input X_0 into two unequal parts such that 9-bit value $X_{0,L}$ is the left part and 7-bit value $X_{0,R}$ is the right part, $X_0 = X_{0,L}||X_{0,R}$. The function includes two non-linear S-boxes S_7 and S_9, S_7 which maps 7-bit input into 7-bit output, and S_9 which maps 9-bit input into 9-bit output.(details of S-boxes are given in appendix) . It also contains two extra functions, f_1 which transforms 7-bit value into 9-bit value by adding two zeros on the left

most side of 7-bit value, and f_2 which transforms 9-bit value into 7-bit value by removing two bits on the left most side of 9-bit value.
The algorithm of the function in round i goes as follows:

$$X_{1,L} = X_{0,R}X_{1,R} = S_9(X_{0,L}) \oplus f_1(X_{0,R}),$$
$$X_{2,L} = X_{1,R} \oplus KI_{i,j,2}X_{2,R} = S_7(X_{1,L}) \oplus f_2(X_{0,R}) \oplus KI_{i,j,2},$$
$$X_{3,L} = X_{2,R}X_{3,R} = S_9(X_{2,L}) \oplus f_1(X_{2,R}),$$
$$X_{4,L} = S_7(X_{3,L}) \oplus f_2(X_{3,R})X_{4,R} = X_{3,R},$$

where $KI_{i,j} = KI_{i,j,1}||KI_{i,j,2}$, $KI_{i,j,2}$ and $KI_{i,j,2}$ are 7-bit and 9-bit values, respectively.

The FO Function: The FO function is a 3-round Feistel construction which mixes 32-bit input data X_0 with a 48-bit subkey KO_i and 48-bit subkey KI_i in round i.

The 32-bit input X_0 is divided into two halves, $X_{0,L}$ and $X_{0,R}$ such that $X_0 = X_{0,L}||X_{0,R}$ and is processed by the following algorithm of the FO function in round i:

For j=1 to 3;
$X_{j,R} = FI(X_{j-1,L} \oplus KO_{i,j}, KL_{i,j}) \oplus X_{j-1,R}$
$X_{j,L} = X_{j-1,R}$
where $KO_i = KO_{i,1}||KO_{i,2}||KO_{i,3}$ and $KL_i = KL_{i,1}||KL_{i,2}||KL_{i,3}$.

Encryption:

Encryption function divides a 64-bit input X_0 into 32-bit halves $X_{0,L}$ and $X_{0,R}$ such that $X_0 = X_{0,L}||X_{0,R}$, then applies the following algorithm:

For i=1,3,5,7;
$X_{i,L} = X_{i-1,L}$ $X_{j,R} = FO(FL(X_{j-1,L}, KL_1), KO_1, KI_1)$
$X_{i+1,L} = X_{i-1,L} \oplus FL(FO(X_{i,R}, KO_{i+1}, KI_{i+1})KL_{i+1}))$ $X_{i+1,R} = X_{i,R}$
The algorithm returns the value $X_8 = X_{8,L}||X_{8,R}$.

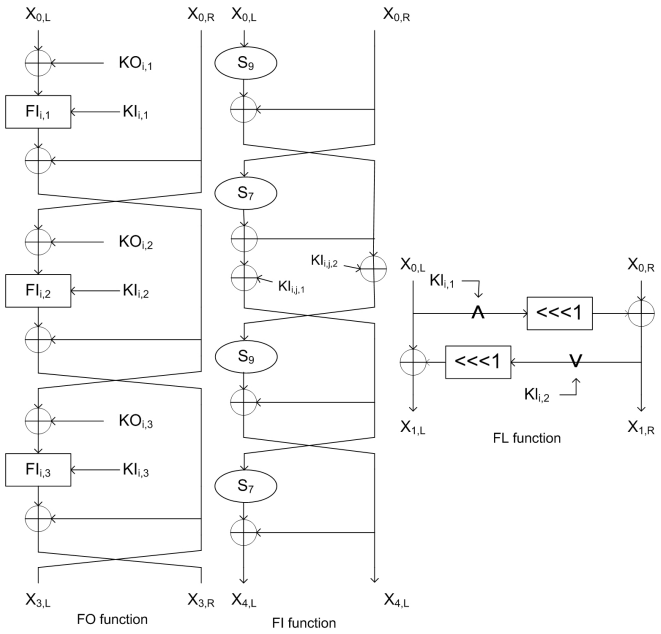

Figure 3.2: Functions of KASUMI

3.2.3 Related-Key Differential Attack on 5 and 6 Rounds KASUMI

3.2.3.1 A Set of 4-Round Differentials for KASUMI

A set of 4-round differentials are built by taking advantage of two properties of FL function which are given below:

Property 1: *If the input to FL function is* $0^{16}1^{16}$*, then the output is always* 1^{32}.

Property 2: *If the input difference to FL function is zero and the subkey differences are* $\Delta KL_{i,2} = 0$ *and* $\Delta KL_{i,1} \neq 0$*, then the output difference is zero with a probability* $2^{-wt(\Delta KL_{i,1})}$*, where* $wt(\Delta KL_{i,1})$ *is the weight of* $\Delta KL_{i,1}$.

The AND of the input bit and two related keys which have difference in the corresponding position of the input bit is always zero when the input bit is zero. However, when the input

bit is 0, after the key mixing, the output will the value 0 and 1, respectively. Hence, the difference of the values after key mixing is zero with a probability 2^{-1}. When the whole input and key strings are considered, the probability of having zero difference between the inputs is $2^{-wt(\Delta KL_{i,1})}$.

If the form of plaintexts is chosen such that $P = (P_L, P_R) = (0^{16}1^{16}, \delta)$ and $P^* = (P_L^*, P_R^*) = (0^{16}1^{16}, \delta \oplus (\gamma <<< 5)0^{16})$, where δ is any 32-bit value and γ is a 16-bit value and the difference between keys K and K^* is taken as $K \oplus K^* = (0, 0, \gamma, 0, 0, 0, 0, 0)$ then the output difference of round 4 will be of the form $\alpha(\gamma <<< 5)0^{16}$ with a probability of $2^{-wt(\gamma)}$. This set of 4-round differentials depicted in Figure 3.3 can be constructed as follows:

- In round 1, the FL function is applied to the left part of the plaintexts $P_L = P_L^* = 0^{16}1^{16}$, by the property 1, the output difference of the function will be zero. Then, zero differences is preserved through the FO function, since the subkey differences ΔKO_1, ΔKI_1 and the input difference to the function FO is zero.

- In round 2, the input difference $((\gamma <<< 5)0^{16})$ to the function FO is canceled by the subkey difference $\Delta KO_{2,1} = (\gamma <<< 5)$, hence the output difference turns out to be zero. Then, the zero difference is preserved through the FL function due to the subkey difference which is zero.

- In round 3, the input difference to the FL function is zero and the subkey difference is $\Delta KL_3 = (\gamma <<< 1)0^{16}$, then the output difference of the function is zero with a probability $2^{-wt(\gamma)}$ by the property 2. Then, zero difference is maintained through the FO function, since the input and the key difference is zero.

- In round 4, the input difference to the FO function is $((\gamma <<< 5)0^{16})$ and the subkey differences are $\Delta KO_4 = 0^{48}$ and $\Delta KI_4 = 0^{32}\gamma$, then the output difference of the function can be any 32-bit value λ. The difference λ leads to an 32-bit difference α through the FL function.

  As a result, the output difference at the end of round 4 becomes $\alpha(\gamma <<< 5)0^{16}$ with probability of $2^{-wt(\gamma)}$, if the plaintext and key differences are taken as above.

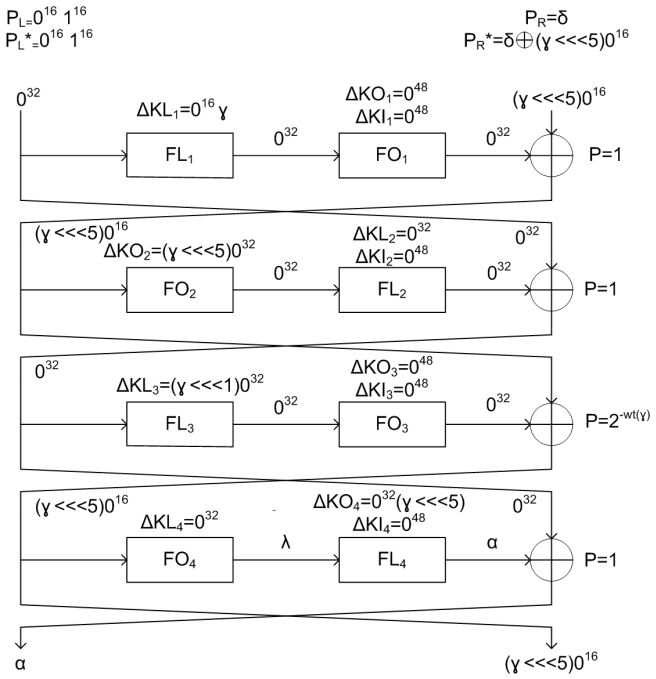

Figure 3.3: A set of 4-round related-key differential characteristics for KASUMI

3.2.3.2 An Attack on 5-Round KASUMI

A 5-round attack [15] is mounted by utilizing one of 4-round differentials mentioned before. In order to perform the attack with lower complexity, the weight of key difference γ is taken as one, that is $wt(\gamma) = 1$. In this way, the probability of a 4-round differential will be $2^{-wt(\gamma)} = 2^{-1}$.

The attack procedure goes as follows:

- Form 2^{15} plaintexts pairs P and P^* such that $P = 0^{16}1^{16}\delta$, $P = 0^{16}1^{16}\delta \oplus (\gamma <<< 5)0^{16}$, where δ is any 32-bit value and γ is a 16-bit value of weight 1.
- Encrypt the plaintext pairs P and P^* under K and K^* and obtain their corresponding plaintexts C and C^*.

  Since 1 bit difference between master keys K and K^* can be in 16 different positions, P^*'s are encrypted under 16 different values of K^*. Therefore, the attack requires 2^{15} plaintext encryptions under key K and $2^{15} \cdot 2^4 = 2^{19}$ plaintext encryptions under key K^*.
- Keep the ciphertext pairs C and C^* satisfying the difference $C \oplus C^* = \alpha ab$, where $a = b \oplus (\gamma <<< 5)$. This step has 16-bit filtering condition, hence, about $2^{-16} \cdot$

$2^{19} = 2^3$ pairs that satisfy the appropriate plaintext and ciphertext differences are left. Since $a = b \oplus (\gamma <<< 5)$, the output difference of the function FO will be $(\gamma <<< 5)0^{16}) \oplus (b \oplus (\gamma <<< 5)b) = bb$. When the difference bb is rolling back through the FO function, it is seen that the input difference to $FI_{5,3}$ will be zero. Because the output difference and the subkey difference of the function is zero. Afterwards, since the subkey difference $\Delta KO_{5,3} = (\gamma <<< 13)$, the output difference of $FI_{5,2}$ will be $b \oplus (\gamma <<< 13)$.

- Guess the subkeys $KL_{5,1}$ and $KO_{5,2}$ and compute the input values to the $FI_{5,2}$ as the ciphertexts are known. Then, for each ciphertext pair, since input differences and output differences of $FI_{5,2}$ are known, by the help of XOR tables of S_7 and S_9,the values of the subkey $KI_{5,2}$ is suggested. Then, store the triples $(KL_{5,1}, KO_{5,2}, KI_{5,2})$ that are suggested at 4 ciphertext pairs. The key triple is uniquely detected due to the fact that an incorrect subkey triple is suggested by 4 ciphertext pairs with a probability of $(2^{-16})^4 = 2^{-64}$ which is less than 2^{-57}. For that reason, a wrong triple is suggested with a probability of less than 2^{-9} and the suggested subkey triple can be accepted as the correct subkey triple. Furthermore, 4 ciphertext pairs suggesting subkey triples can be considered as the right pairs that follows the differential path prescribed before.

- Once having obtained the subkey $KI_{5,2}$, the right part of the input difference to the FO function is known and denote the difference c. Then, the output difference of $FI_{5,1}$ is $(\gamma <<< 13) \oplus c)$, and by guessing the subkeys $KL_{5,2}$ and $KO_{5,1}$ which enables the input to $FI_{5,1}$ to be attained for each right pair. Since the input difference and the output difference of $FI_{5,1}$ are known, the subkey $KI_{5,1}$ can be recovered by the same method used in the previous step.

- Up to now, 96 bits of the original key are recovered. The remaining bits can be found by exhaustive search or forming another set of plaintext-ciphertext pairs which satisfy the condition stated in the first step and do not satisfy the condition in the third step.

The time complexity of the attack is about 2^{33} 5-round KASUMI encryptions and the data complexity of attack is 2^{15} and 2^{19} chosen plaintexts which are encrypted under master keys K and K^*.

3.2.3.3 An Attack on 6-Round KASUMI

A 6-round attack can be performed by utilizing the same differential used in the 5-round attack. When mounting this attack, it is avoided guessing all subkey bits of round 6 to peel off the round by choosing proper differences between the ciphertext pairs. The attack can be described as follows:

- As in the previous 5-round attack, form $3 \cdot 2^{13}$ plaintexts pairs P and P^* such that $P = 0^{16}1^{16}\delta$, $P = 0^{16}1^{16}\delta \oplus (\gamma <<< 5)0^{16}$, where δ is any 32-bit value and γ is a 16-bit value of weight 1.
- Ask the encryptions of the plaintext pairs under P and P^* under K and K^* and obtain their corresponding plaintexts C and C^*, respectively.
  Since K^* has 16 different values due to the position of one bit difference, the attack requires $3 \cdot 2^{13}$ plaintext encryptions under key K and $3 \cdot 2^{17}$ plaintext encryptions under key K^*.
- Keep the ciphertext pairs C and C^* such that each pair satisfies the difference $C \oplus C^* = cab$, where $b = (\gamma <<< 8)$. There is 16-bit filtering condition on ciphertext pairs, therefore, the number of remaining pairs that satisfy the appropriate plaintext and ciphertext differences are about $2^{-16} \cdot 3 \cdot 2^{17} = 6$.
  If the difference ab propagates through FL_6 function, it is observed that the difference $b = (\gamma <<< 8)$ is cancelled by the subkey difference $\Delta KO_{6,2} = (\gamma <<< 8)$ which makes the input difference to $FI_{6,2}$ zero. Since $\Delta KI_{6,2} = 0$, the output difference of $FL_{6,2}$ is also zero.
- To peel off round 6, guess 6 subkeys of round 6 $KL_{6,1}$,$KL_{6,2}$,$KO_{6,1}$,$KO_{6,3}$,$KI_{6,1}$ and $KI_{6,3}$ which are in fact 96 bits of the master key K. Note that it is not necessary to guess subkey bits of $KO_{6,2}$ and $KI_{6,2}$, since the 16-bit output value of $FI_{6,2}$ can be known by considering 2^{16} possibilities for it.

  Note that if the subkeys of round 6 specified above are guessed, then the subkeys of round 5 $KL_{5,1}$,$KL_{5,2}$,$KO_{5,1}$,$KO_{5,2}$,$KI_{5,2}$ and $KI_{5,3}$ will be determined due to the feature of the key schedule of KASUMI. Therefore, only unknown subkeys of round 5 are $KI_{5,1}$ and $KO_{5,3}$.
- By guessing the subkeys and considering each possible value of the output $FI_{6,2}$,

round 6 is passed and the input values of FL_5 are known for each ciphertexts.

- As input values of $FI_{5,2}$ are known and all bits of $KO_{5,2}$ and $KI_{5,2}$ are previously guessed in round 6 , the output values of $FI_{5,2}$ are also known. Now, by using each suggested pair $KI_{5,1}$ and $KO_{5,3}$ and 96 guessed subkey bits, find the output difference of FL_4 for a pair of ciphertexts. Then, keep the subkey pair $KO_{5,2}$ and $KI_{5,2}$ which makes the output difference of FL_4 be equal to the input difference of FL_5. To prevent double counting, if this subkey pair is not already suggested by the same ciphertext pair for a different output value of $FI_{6,2}$ and same value of 96 key bits., insert this value to the list of suggested values of this pair. If it is previously written in the list, then do not add it to the list again.
- Since the probability of the differential characteristics is $\frac{1}{2}$, the expected number of remaining pairs is $6 \cdot \frac{1}{2} = 3$. So, the correct values of the subkeys $KL_{6,1}$, $KL_{6,2}$, $KO_{6,1}$, $KO_{6,2}$, $KO_{6,3}$, $KI_{6,1}$, $KI_{6,2}$ and $KI_{6,3}$ are the value that are suggested at least three times.

The data complexity of the attack is $(3 \cdot 2^{13}) + (3 \cdot 2^{17})$ related-key chosen plaintexts and the time complexity of the attack is $1.5 \cdot 2^{113}$ 6-round KASUMI encryptions.

CHAPTER 4

RELATED-KEY IMPOSSIBLE DIFFERENTIAL CRYPTANALYSIS

This chapter is mainly about the related-key impossible differential cryptanalysis which is a combination of the related-key cryptanalysis and the impossible differential cryptanalysis. It is known to be a valuable attack that have been applied to several block ciphers.

Impossible differential cryptanalysis was independently presented by Knudsen [21] in 1998 and Biham et al. [22] in 1999 and it was firstly applied to DEAL and Skipjack by Knudsen and Biham, respectively. It exploits impossible differential relations between plaintext and ciphertext pairs, that is it uses a differentials with probability 0. The attack is mostly constructed in miss in the middle manner [23]; that is, two differentials with probability 1 are concatenated in a way that intermediate differences contradict with each other. Then, the idea of the attack is combined with the related-key idea such that the attacker uses differentials which hold with probability 0 and requires an assumption that the attacker knows the particular differences between one or more pairs of unknown keys.

The overview of this chapter is as follows: In Section 6.1, a brief description of the related-key impossible differential cryptanalysis is made. In Section 4.2, related-key impossible differential attack on reduced round AES proposed by E. biham is discussed. Then in Section 4.3, a related-key impossible differential attack on 31-round HIGHT proposed by Özen et al. in 2009 is given [24]. Finally, in Section 4.4, our 25-round related-key impossible distinguisher for XTEA is introduced.

4.1 Overview of the Attack

Let α be an input difference and β be an output difference, the related-key impossible differential is that the input difference α never propagates the output difference β under related keys K and K', simply written $\alpha \nrightarrow \beta$ with probability 1 ,in other words, α difference cannot produce β difference under any key.

The related-key differential with probability 0 can be used as a distinguisher for a cipher, because for a random mapping the input difference α leads to an output difference β with probability of 2^{-n} under related-keys. So, by taking a sufficient number of plaintext-ciphertext pairs, the cipher can be differentiated from a random mapping.

To describe the related-key impossible differential attack, consider the block cipher $E : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ as a cascade of three subciphers $E = E^f \circ E^m \circ E^i$, where E^m is rounds including the impossible differential, E^i is the initial rounds before E^m and E^f is rounds after E^m. Then, build a related-key impossible differential distinguisher that satisfies the following equation:

$$Prob[E^m_K(P) \oplus E^m_{K'}(P \oplus \alpha) = \beta] = 0 \tag{4.1}$$

where, P and $P \oplus \alpha$ are plaintexts, K and K' are related keys, and α and β are plaintext and ciphertext differences, respectively.

Using a related-key impossible differential distinguisher, the key recovery attack can usually be constructed as follows:

- Guess necessary key bits/bytes in E^i, partially encrypt the plaintext pairs under guessed keys K and K' and take the pairs that satisfy the difference α just before E^m. Then, encrypt them under keys K and K' and get their corresponding ciphertexts.

- Guess necessary key bits/bytes in E^f, partially decrypt the ciphertext pairs under guessed keys K and K' and take the pairs that satisfy the difference β just after E^m.
- A sufficient number of plaintext-ciphertext pairs that satisfy the two conditions specified above eliminates wrong keys eliminate and leave the actual key.

4.2 Related-Key Impossible Differential Cryptanalysis of Reduced-Round AES

AES is designed to resist both differential and linear cryptanalysis. J. Daemen and V. Rijmen, the designers of Rijndael, proved in the AES proposal [25] that there are no 4 or more rounds differential trails to mount differential attack due to the diffusion property of mix column operation. However, it is observed that the key scheduling algorithm of AES is not as strong as AES encryption algorithm and propagates differences slowly. This property of the key scheduling algorithm makes the cipher vulnerable to related-key attacks. For this reason, in recent years, several related-key impossible differential attacks on AES have been presented. In 2003, Jakimoski and Desmedt [26] proposed related key impossible differential attacks on 7 and 8 rounds of AES-192. Then in 2006, Biham et al. [27] mounted attacks on again 7 and 8-round AES-192 following the work in [26] but with an improvement of the data and time complexities. In [28], Zhang et al. presented related-key impossible differential attacks different from the ones in [26] and [27], by choosing another key difference between two related keys and starting the attack from the first round which improved the data and time complexities. Moreover, Zhang et al. [29] made a security analysis of AES-256 against the related-key impossible differential attack and they introduced an attack on 7-round AES-256 and four attacks on 8-round AES-256.

Note that the security of AES-192 against related key attacks has inspired much research because of being more susceptible to related key attacks than other key variants. The reason is that AES-192 has longer related key differentials since the diffusion of the key schedule is slower than in other versions. More specifically, if the expanded keys are thought as a sequence of words, then the key schedule of AES-192 applies a non-linear transformation once every six words, while the key schedules of AES-128 and AES-256 apply non-linear transformations once every four words.

In this section, the best related-key impossible attack on reduced-round AES-192 presented by Zhang et al. [28] will be covered. To perform the attack, the encryption algorithm and the key scheduling algorithm of AES-192 is analyzed, appropriate differences between two related keys are chosen and then 5.5-round related-key impossible differential which combines two related-key differentials in opposite directions giving rise to a contradiction is constructed.

4.2.1 Notations

The notations used in the description of AES and attack are given as follows:

k_i: The subkey of round i, $0 \leq i \leq 3$

$k_{i,Col(j)}$: The j^{th} column of k_i, $0 \leq i \leq 3, 0 \leq j \leq 3$

x_i^I: The input to round i

x_i^S: The output of SubByte operation in round i

x_i^R: The output of ShiftRow operation in round i

x_i^M: The output of MixColumn operation in round i

x_i^O: The output of round i

$x_{i,col(j)}$: The j^{th} column of x_i , $0 \leq i \leq 3, 0 \leq j \leq 3$

$(x_i)_j$: The j^{th} byte of x_i, $0 \leq i \leq 3, 0 \leq j \leq 3$

?: Any byte difference

4.2.2 The AES Block Cipher

Rijndael [25] is block cipher which was designed by Rijmen and Daemen is the winner of The Advanced Encryption Standard competition organized by National Institute of Standards and Technology (NIST) in 2001. Then, it is called the Advanced Encryption Standard (AES). AES has replaced for aging DES and has recently become the most widely used block cipher in the world.

AES has an 128-bit SPN structure with three different key sizes: AES-128, AES-192 and AES-256. According to its key leght, each AES version uses different key schedules. However, this thesis only includes the attacks on AES-192, only give the key schedule of AES-192 is described. In addition, each version has different number of rounds, AES-128, AES-192

and AES-256 have 10,12 and 14 rounds, respectively.

The round function of AES consists of four operations:

SubByte (SB): It is a non-linear operation on $4x4$ arrays and includes an $8x8$ bijective S-box which is applied all 16 bytes of internal state bytes. For the details of the S-box, refer to [25].

ShiftRow (SR): It is linear function which applies cyclic shift to ith row of a $4x4$ array to the left by i bytes, where $0 \leq i \leq 3$.

MixColumn (MC): It is a permutation that multiply each column by a constant $4x4$ matrix over the Galois Field $GF(2^8)$ where M is defined as:

$$\text{M}=\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

AddRoundKey (AR): It mixes the state and a 16 byte subkey by using bitwise XOR operation. Namely, if A and B are 16-byte subkey and internal states, respectively, then $AR(A, B) = A \oplus B$.

The Key Schedule of AES-192

The key schedule of AES-192 generates thirteen 128-bit (16-byte) subkeys by using 192-bit secret key. The algorithm to generate 16-byte subkeys by using the following algorithm:

1. Divide 192-bit user key into six 32-bit words $(K_1, K_2, \ldots K_6)$
2. For $j = 7$ to $j = 52$, do
   If $(j \equiv 1 mod 6)$, then $K_i = K_{j-6} \oplus SB(K_j <<< 8) \oplus RCON[i/6]$
   Else $K_i = K_{i-6} \oplus K_{i-1}$
3. Output $k_i = (K_{4i+1}, K_{4i+2}, K_{4i+3}, K_{4i+4})$, $(0 \leq i \leq 12)$

Encryption Algorithm of AES-192

Let denote 128-bit plaintext and ciphertext blocks as P and C, respectively. The encryption Algorithm of AES-192 can be described as follows:

- Input Plaintext P,
- $A_0 = AR(P, k_0)$,
- For $i = 0$ to 11, do
- $B_i = SB(A_{i-1})$
  $C_i = SR(B_i)$
  $D_i = MC(C_i)$
  $A_i = AR(D_i, k_i)$
- $A_{12} = SB(A_{11})$
  $B_{12} = SR(A_{12})$
- Ciphertext $C = AR(B_{12}, k_{12})$

Where A_i, B_i, C_i and D_i are $4x4$ byte arrays.

4.2.3 Some Properties of AES-192

Property 1: *If two inputs to MC operation differ in only one byte in the ith column, then the outputs of the MC operation differ in four (all) bytes of the ith column.*

This is a direct consequence of the description of MC operation.

Property 2: *If two inputs to MC operation differ in any two bytes in the ith column, then the output difference of MC operation cannot be determined.(It can be zero difference or non-zero difference(s) in any byte(s))*

This is also a direct consequence of the description of MC operation.

Property 3: *If the key differences are chosen as follows*:

$(\Delta K_{col(0)}, \Delta K_{col(1)}, \ldots \Delta K_{col(5)})$=((a,0,0,0), (0,0,0,0), (a,0,0,0), (0,0,0,0), (0,0,0,0), (0,0,0,0))

then the subkey differences are used in the attack are given in Table 5.4:

Table 4.1: Subkey Differences

Round(i)	$\Delta k_{i,Col(0)}$	$\Delta k_{i,Col(1)}$	$\Delta k_{i,Col(2)}$	$\Delta k_{i,Col(3)}$
0	(a,0,0,0)	(0,0,0,0)	(a,0,0,0)	(0,0,0,0)
1	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
2	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
3	(a,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
4	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
5	(a,0,0,0)	(a,0,0,0)	(a,0,0,0)	(a,0,0,0)
6	(a,0,0,b)	(0,0,0,b)	(a,0,0,b)	(0,0,0,b)
7	(a,0,0,b)	(0,0,0,b)	(a,0,c,b)	(a,0,c,0)
8	(0,0,c,b)	(0,0,c,0)	(a,0,c,b)	(a,0,c,0)

4.2.4 A 5.5-Round Related Key Impossible Differential of AES-192

A 5.5-round related-key impossible differential which starts with the output of the MC operation of round 1 and ends with the output of round 6 is:

$$((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0)) \nrightarrow$$
$$((?,?,?,?),(?,?,?,?),(?,?,?,?),(0,0,0,b))$$

This related-key impossible differential is built by combining two related-key differentials: a 4.5-round related-key differential in encryption direction and a 1-round related-key differential in decryption direction such that intermediate differences of these two differentials give rise to a contradiction.

The First 4.5-Round Differential:

A 4.5-round related-key differential is depicted in Figure 4.1 is built as follows:

- Begin by giving the difference (0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0) to the output of the MC operation of round 1 which is cancelled by the subkey difference of round 1, so the difference in all bytes of output of round 1 becomes zero.

- This zero output difference of round 1 is kept through SB,SR MC and AR operations of round 2 and SB,SR MC operations of round 3.

- Then, at the end of round 3, the output difference becomes ((a,0,0,0), (0,0,0,0), (0,0,0,0), (0,0,0,0)) due to the addition of the subkey difference of round 3.

- This non-zero byte difference diffuses to all bytes of 0-th column due to the property 1 and at the end of round 4, the output difference becomes ((N,N,N,N),(0,0,0,0), (a,0,0,0), (a,0,0,0)).

- After applying SB and SR operations of round 5, the difference becomes ((N,0,0,0), (0,0,0,N), (N,0,N,0), (N,N,0,0)), then due to property 2, the output of MC operation is ((N,N,N,N), (N,N,N,N), (?,?,?,?), (?,?,?,?)).

- Finally, after adding the subkey of round 5, the difference ends up with ((?,N,N,N), (?,N,N,N), (?,?,?,?), (?,?,?,?)).

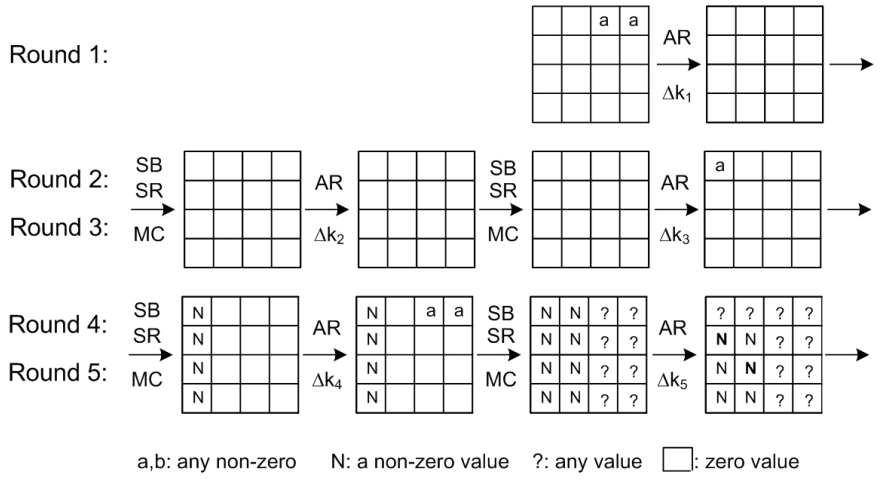

Figure 4.1: A 4.5-round related-key differential for AES-192

This completes the first related-key differential.

The Second 1-Round Differential:

A 1-round related-key differential shown in Figure 4.2 is built as follows:

- Give the difference ((?,?,?,?), (?,?,?,?), (?,?,?,?), (0,0,0,b)) to the output of round 6.

- Then, go backward through AR, MC, SR and SB operations and get the difference ((?,0,?,?), (?,?,0,?), (?,?,?,0), (0,?,?,?)) as the input difference to round 6.

This completes the second related-key differential.

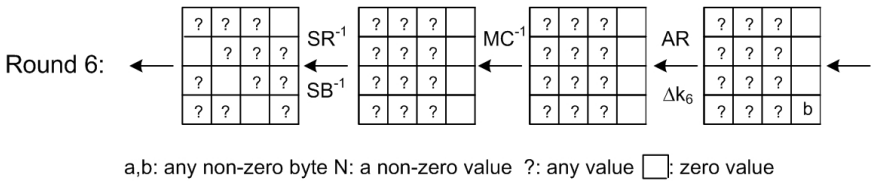

Figure 4.2: A 1-round related-key differential for AES-192

Constructing a 5.5 round related-key impossible differential distinguisher:

To construct the distinguisher, two differentials given above are concatenated as in Figure 4.3.

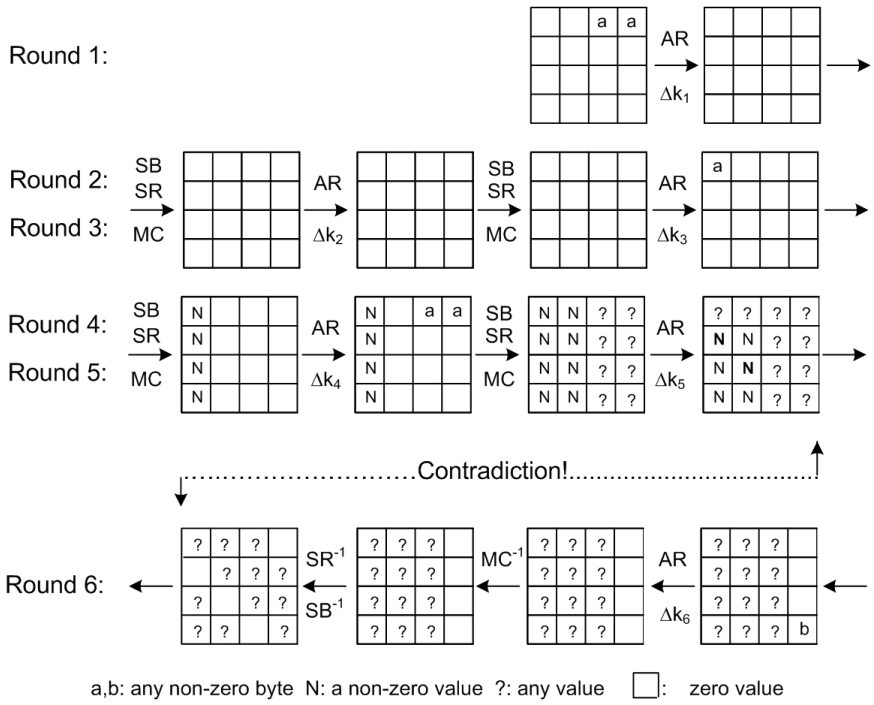

Figure 4.3: A 5.5 round related-key impossible differential distinguisher for AES-192

The output difference of round 5 must be equal to the input difference of round 6, however, it is seen that the output difference of the first byte of round 5 $(\Delta x_5^O)_1$ which is non-zero is not equivalent to the input difference of the first byte of round 6 $(\Delta x_6^I)_1$ which is zero. This gives rise to a contradiction.

4.2.5 A 7-Round Related-Key Impossible Differential Attack on AES-192

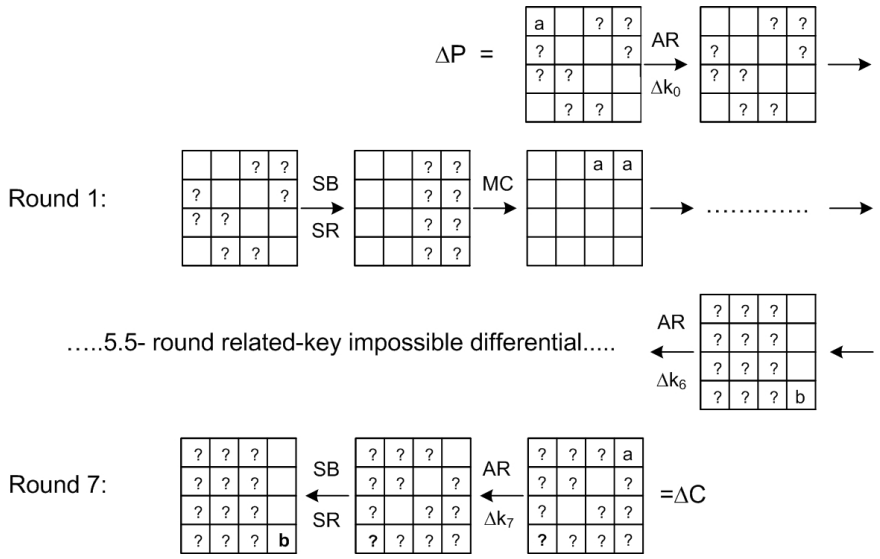

Figure 4.4: Rounds before and after the related-key impossible distinguisher for AES-192

The aim of the attack is to capture 8 bytes of k_0 and the third byte of k_7. Firstly, it is assumed that the values of a,b,c are all known and all possibilities of these values will be considered in the part of complexity of the attack.

The attack algorithm goes as follows:

First of all, to make attack faster, the authors construct a hash table storing the input differences of possible plaintext that satisfy the input difference to the related-key impossible differential.

Precomputation step: Form all 2^{64} possible values that have the difference (a,0,0,0) in their second and third columns and roll back the values through MC, SR and SB operations of round 1. Then, calculate the values of eight bytes 1,2,6,7,8,11,12 and 13 of the input to the

first round and put the values in to a hash table indexed by the difference in these bytes.

The Attack:

- Form two sets of plaintexts $\mathcal{P}_1$ and $\mathcal{P}_2$ containing m plaintexts each such that each pair $P_1 \in \mathcal{P}_1$ and $P_2 \in \mathcal{P}_2$ satisfies the desired difference =((a,?,?,0),(0,0,?,?),(?,0,0,?),(?,?,0,0)).

- Ask encryption of the sets of plantexts $\mathcal{P}_1$ and $\mathcal{P}_2$ and obtain the sets of their corresponding ciphertexts C_1 and C_2 under two related keys K and $K^{'}$, respectively.

- Construct the set $C_2^* = \{C_2{}^* | C_2 \in C_2\}$ by computing $C_2{}^* = C_2 \oplus$((0,0,0,0), (0,0,0,0), (0,0,0,0), (a,0,0,0)) for every ciphertexts $C_2 \in C_2$

- Build a hash table storing the the ciphertexts $C_2 \in C_2$ and $C_2^* \in C_2^*$ indexed by the bytes 6,9,12.

- Then, by guessing the value of the third byte of k_7, do the following steps:

    1. First of all, list all 2^{64} values of the subkey k_0 in the bytes 1,2,6,7,8,11,12 and 13.
    2. Decrypt the third byte of all ciphertext through AR, SR and SB operations of round 7 .
    3. Keep the pairs C_2 and C_2^* which are in the same bin of the hash table having the difference 'b' in their third bytes of input to round 7.
    4. For every remaning pairs C_2 and C_2^*, look at their corresponding plaintexts P_1 and P_2 and calculate the difference in their bytes 1,2,6,7,8,11,12 and 13 and call the difference value as Δ.
    5. Finally, access the bin Δ in the hash table, if this is nonempty, for each pair $(P, P^{'})$ calculate the values $P_1 \oplus P$ and $P_1 \oplus P^{'}$ and remove these values from the list of 2^{64} values of the subkey k_0.
    6. The remaining value in the list of all 2^{64} possible values of k_0 and the guessed value of $(k_7)_3$ are the actual subkey bytes.

4.2.6 Attack Complexity

Data Complexity: Since two sets of plaintexts contain m plaintexts each, there are m^2 possible ciphertexts pairs (C_1, C_2^*). There is a filtering condition in item 4; so, the remaining ciphertext pairs in the same bin of the hash table is $2^{-24}m^2$. Then, in item 5(3), there is a 8-bit filtering condition and 2^{-8} of ciphertext pairs are eliminated for one guess of subkey $(k_7)_3$. So, there remain about $2^{-32}m^2$ ciphertext pairs. Also, it is expected that each plaintext-ciphertext pair satisfying related-key impossible differential eliminates one wrong key. Since for one guess of $(k_7)_3$, there are 2^{64} subkey candidates, each pair eliminates 2^{-64} of wrong key values, therefore after the first pair there remain $2^{64}-1 = 2^{64}(1-2^{-64})$ keys. After the second pair, there remain $2^{64}(1-2^{-64}) - 2^{64}(1-2^{-64})2^{-64} = 2^{64}(1-2^{-64})^2$ keys. Then, after kth pair, it is expected to have $2^{64}(1-2^{-64})^k$ keys. The aim is to make the inequality $2^{64}(1-2^{-64})^k < 1$ hold to eliminate all wrong subkey values. So, it is enough to take 2^{70} pairs to make only the right subkey leave. If $k = 2^{70}$, the number of remaining wrong keys become $2^{64}(1-2^{-64})^{70} \approx 2^{-28.85}$ which is less than 1. To get 2^{70} pairs, it is needed to take $m = 2^{51}$ related-key chosen plaintext-ciphertext pairs for each two sets. Therefore, the data complexity of the attack is $2 \cdot 2^{51} = 2^{52}$ related-key chosen plaintexts.

Time Complexity: The time complexity of the attack is dominated by item 5(5). In item 5(5) 2^{70} pairs are dealt with and for each pair, it is needed to perform one memory access to hash table and the list values of k_0. Since this step is done for each guess of $(k_7)_3$, the time complexity of this step is $2 \cdot 2^8 \cdot 2^{70} = 2^{79}$ memory access which is equal to 2^{73} AES encryptions. Furthermore, precomputation step needs about $2 \cdot 2^{64} \cdot \frac{1}{7} \approx 2^{62}$ encryptions.

In addition, in the attack, it is assumed that the values of a,b and c are known, however , the attacker can fix only the value of a to a certain value and should give all possible values to b and c. Since there is no need to know the value of c, the attack should be repeated for all possible 2^7 values of b and only the time complexity of the attack should be multiplied by 2^7. Therefore, the time complexity of the attack is $2^7 \cdot 2^{72} \approx 2^{80}$ AES encryptions.

Required Memory: The attack needs 2^{69} bytes memory which is dominated by precomputation step.

4.3 Related-Key Impossible Differential Cryptanalysis of 31-Round HIGHT

Nowadays, light-weight block ciphers gain importance in some areas of real-life cryptographic applications, such as Radio frequency identification (RFID)systems or Ubiquitous Sensor Networks (USN). To use in these kind of applications, designing cryptograhic algorithms becomes as an important issue. Therefore, HIGHT is proposed by Hong et al. in 2006 to meet the demand on block ciphers which is used in this area [30]. The design goal of HIGHT is to provide low-resource hardware implementation. So, the significant property of HIGHT is that it is composed of only simple algebraic operations such as bitwise XOR, addition in modulo 2^8 and left bit rotations. In addition, the security of HIGHT is analyzed in [30] and it is proved that HIGHT is resistant to some known attacks such as linear, differential, truncated differential, impossible differential, saturation and boomerang attacks.

In the proposal of HIGHT, designers claimed that the best impossible differential attack can be applied to 18-round HIGHT by using 14-round impossible differential. However, an impossible differential attack to 25-round HIGHT and a related-key impossible differential attack to 28-round HIGHT is proposed by Lu in 2007 [31]. Then in 2008, Varici et al. further analyzed HIGHT and increased the number of attacking rounds from 28 to 31 by using 22-round related-key impossible differential. The complexity of this attack is approximately same as exhaustive search, however, one can conclude that HIGHT is not resistant against impossible differential attacks as claimed in the proposal.

In this section, the related-key impossible differential attack on 31-round HIGHT [24] will be covered.

4.3.1 Notations

The notations used in the description of HIGHT and the attack are given as follows:

$\oplus$: Exclusive OR operation
$\boxplus$: Addition in modulo 2^8

$<<< i$: i-bit rotation to the left

K: A 128-bit master key

K_i: ith byte of master key K, $0 \leq i \leq 15$

WK_i: ith byte of whitening key, $0 \leq i \leq 7$

$X_{i,j}$: jth input byte to the round i, $0 \leq i \leq 31$ and $0 \leq j \leq 7$

e_i: A byte having zeros in all positions except the position i

$e_{i,\sim}$: A byte having zeros in bit positions 0 to $i-1$, one in ith position, and any values in positions $i+1$ to 7.

$e_{\bar{i},\sim}$: A byte having zeros in bit positions 0 to i and any values in positions $i+1$ to 7.

?: Any byte difference

Denote the 64-bit plaintext P as a sequence of 8 bytes $(P_7, P_6, \ldots, P_0)$ and 64-bit ciphertext C as a sequence of 8 bytes $C = (C_7, C_6, \ldots, C_0)$.

4.3.2 The HIGHT Block Cipher

HIGHT is a light-weight block cipher with 64-bit block size and 128-bit key size. It has an iterative 32-round structure which is Feistel-type with 8 branches.

4.3.2.1 Key Scheduling Algorithm

A 64-bit user key $K = (K_{15}, K_{14}, \ldots K_0)$ is used to generate 8 whitening key bytes $WK_0, WK_1, \ldots WK_7$ and 128 subkey bytes $SK_0, SK_1, \ldots SK_{127}$.

Whitening keys are generated as follows:

$$\begin{aligned} WK_j &= K_{j+12}, j = 0, 1, 2, 3, \\ WK_j &= K_{j-4}, j = 4, 5, 6, 7. \end{aligned}$$

The whitening key bytes WK_0, WK_1, WK_2 and WK_3 are used in the input whitening part and WK_4, WK_5, WK_6 and WK_7 are used in the output whitening part of the encryption algorithm of HIGHT.

The algorithm that produces round subkeys is given as follows:

$$\begin{aligned} SK_{16i+j} &= K_{(j-i) mod 8} \boxplus \delta_{16i+j}, \\ SK_{16i+j+8} &= K_{(j-i) mod 8+8} \boxplus \delta_{16i+j+8}. \end{aligned}$$

where δ_{16i+j} and $\delta_{16i+j+8}$ are publicly known constant values which were generated by a linear feedback shift register.

4.3.2.2 The Encryption Function of HIGHT

Encryption function of HIGHT consists of three stages, namely initial transformation, round function and final transformation.

Initial Transformation: Initial transformation or output whitening step converts an 64-bit plaintext $P = (P_7, P_6, \ldots, P_0)$ into an 64-bit input value to the first round function:

$$(X_{0,7}, X_{0,6}, \ldots X_{0,0}) = (P_7, P_6 \oplus WK_3, P_5, P_4 \boxplus WK_2, P_3, P_2 \oplus WK_1, P_1, P_0 \boxplus WK_0)$$

Round Function: The round function of HIGHT depicted in figure 4.5 divides an 64-bit input into 8 bytes and then applies operations ($\oplus$ and $\boxplus$ operations) and cyclic rotations to four of these bytes. In other words, the round function of HIGHT only operates on only half of its input bytes.

In the encryption function of HIGHT, the round function is repeated 32-times and can be described as follows:

For $i = 1$ to 32:

$X_{i,0} = X_{i-1,7} \oplus (F_0(X_{i-1,6}) \boxplus SK_{4i-1})$

$X_{i,1} = X_{i-1,0}$

$X_{i,2} = X_{i-1,1} \boxplus (F_1(X_{i-1,0}) \oplus SK_{4i-2})$

$X_{i,3} = X_{i-1,2}$

$X_{i,4} = X_{i-1,3} \oplus (F_0(X_{i-1,2}) \boxplus SK_{4i-3})$

$X_{i,5} = X_{i-1,4}$

$X_{i,6} = X_{i-1,5} \boxplus (F_1(X_{i-1,4}) \oplus SK_{4i-4})$

$X_{i,7} = X_{i-1,6}$

In the above algorithm F_0 and F_1 are two linear functions each of which applies only cyclic rotations to the left to its 8-bit input values and can be defined as follows:

$F_0 : \{0,1\}^8 \rightarrow \{0,1\}^8$, $F_0(x)$=$(x <<< 1) \oplus (x <<< 2) \oplus (x <<< 7)$,

$F_1 : \{0,1\}^8 \rightarrow \{0,1\}^8$, $F_1(x)$=$(x <<< 3) \oplus (x <<< 4) \oplus (x <<< 6)$,

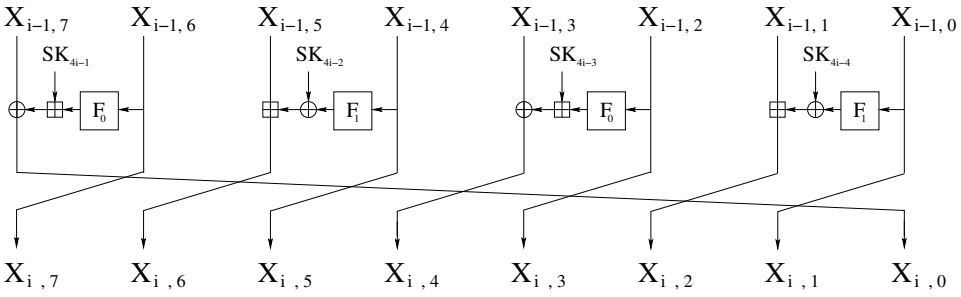

Figure 4.5: ith round function of HIGHT

Final Transformation: Final transformation or output whitening step transforms the output of 32th round function $X_{32} = (X_{32,7}, X_{32,6}, \ldots, X_{32,0},)$ into ciphertext $C = (C_7, C_6, \ldots, C_0)$ as follows:

$$\begin{aligned} C &= (C_7, C_6, \ldots, C_0) \\ &= (X_{32,0}, X_{32,7} \oplus WK_7, X_{32,6}, X_{32,5} \boxplus WK_6, X_{32,4}, X_{32,3} \oplus WK_5, X_{32,2}, X_{32,1} \boxplus WK_4) \end{aligned}$$

4.3.3 A 31-Round Related-Key Impossible Differential Attack on HIGHT

The attack is performed by taking advantage of some properties of HIGHT enumerated below:

1. Properties of the addition operation $\boxplus$ and the Exclusive OR operation $\oplus$ used in the

round function of HIGHT:

- If the $\boxplus$ operation is applied to two pairs each of which have difference in the most significant bit, than the resulting difference will be 0.
- If two inputs to the $\boxplus$ operation differ firstly in the ith bit, then the output difference is preserved in bits 0 up to ith, that is the difference never propagates to the bits in the least significant positions.
- The input difference is always preserved through the $\oplus$ operation.

2. The property of Feistel structure of HIGHT:

   - An input byte difference to ith round will influence only at most two bytes of the output of ith round.

3. The property of the key scheduling algorithm of HIGHT:

   - The diffusion in the key schedule of HIGHT is not satisfied well, because, it can be seen in Table 4.2 which shows the relations between the master key bytes and subkey and whitening key bytes that if a difference is given to any byte of master key K, then only one byte of whitening keys and 8 byte of round subkeys will be affected.

Table 4.2: Relations Between the Master Key, Whitening Keys and Round Keys

Master Key	Whitening Key	Subkey							
K_{15}	WK_3	SK_{15}	SK_{24}	SK_{41}	SK_{58}	SK_{75}	SK_{92}	SK_{109}	SK_{126}
K_{14}	WK_2	SK_{14}	SK_{31}	SK_{40}	SK_{57}	SK_{74}	SK_{91}	SK_{108}	SK_{125}
K_{13}	WK_1	SK_{13}	SK_{30}	SK_{47}	SK_{56}	SK_{73}	SK_{90}	SK_{107}	SK_{124}
K_{12}	WK_0	SK_{12}	SK_{29}	SK_{46}	SK_{63}	SK_{72}	SK_{89}	SK_{106}	SK_{123}
K_{11}	WK_{15}	SK_{11}	SK_{28}	SK_{45}	SK_{62}	SK_{79}	SK_{88}	SK_{105}	SK_{122}
K_{10}	WK_{14}	SK_{10}	SK_{27}	SK_{44}	SK_{61}	SK_{78}	SK_{95}	SK_{104}	SK_{121}
K_9	WK_{13}	SK_9	SK_{26}	SK_{43}	SK_{60}	SK_{77}	SK_{94}	SK_{111}	SK_{120}
K_8	WK_{12}	SK_8	SK_{25}	SK_{42}	SK_{59}	SK_{76}	SK_{93}	SK_{110}	SK_{127}
K_7	WK_{11}	SK_7	SK_{16}	SK_{33}	SK_{50}	SK_{67}	SK_{84}	SK_{101}	SK_{118}
K_6	WK_{10}	SK_6	SK_{23}	SK_{32}	SK_{49}	SK_{66}	SK_{83}	SK_{100}	SK_{117}
K_5	WK_9	SK_5	SK_{22}	SK_{39}	SK_{48}	SK_{65}	SK_{82}	SK_{99}	SK_{116}
K_4	WK_8	SK_4	SK_{21}	SK_{38}	SK_{55}	SK_{64}	SK_{81}	SK_{98}	SK_{115}
K_3	WK_7	SK_3	SK_{20}	SK_{37}	SK_{54}	SK_{71}	SK_{80}	SK_{97}	SK_{114}
K_2	WK_6	SK_2	SK_{19}	SK_{36}	SK_{53}	SK_{70}	SK_{87}	SK_{96}	SK_{113}
K_1	WK_5	SK_1	SK_{18}	SK_{35}	SK_{52}	SK_{69}	SK_{86}	SK_{103}	SK_{112}
K_0	WK_4	SK_0	SK_{17}	SK_{34}	SK_{51}	SK_{68}	SK_{85}	SK_{102}	SK_{119}

4.3.3.1 A 22-Round Related-Key Impossible Differential for HIGHT

A related-key impossible differential presented in [24] which occurs between the rounds 7 and 30 is as follows:

$$(0,0,0,0,0,0,80_x,0) \nrightarrow (0,0,0,80_x,0,0,0,0)$$

under the key difference:

$$(\Delta K_{15}, \Delta K_{14}, \ldots, \Delta K_0) = (80_x,0,0,0,0,0,0,0)$$

The contradiction occurs between the third bytes of output of round 17 and input of round 18. The intermediate differences and the subkey differences of the distinguisher are given in Table 4.3.3.1.

Table 4.3: A 22-Round Related-Key Impossible Differential for HIGHT

	$\Delta X_{i-1,7}$	$\Delta X_{i-1,6}$	$\Delta X_{i-1,5}$	$\Delta X_{i-1,4}$	$\Delta X_{i-1,3}$	$\Delta X_{i-1,2}$	$\Delta X_{i-1,1}$	$\Delta X_{i-1,0}$	Subkey differences			
ΔX_6	0	0	0	0	0	0	80_x	0	0	0	0	80_x
ΔX_7	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_8	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_9	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_{10}	0	0	0	0	0	0	0	0	0	0	80_x	0
ΔX_{11}	0	0	0	80_x	0	0	0	0	0	0	0	0
ΔX_{12}	0	$e_{2,\sim}$	80_x	0	0	0	0	0	0	0	0	0
ΔX_{13}	$e_{2,\sim}$	80_x	0	0	0	0	0	?	0	0	0	0
ΔX_{14}	80_x	0	0	0	0	?	?	$e_{0,\sim}$	0	80_x	0	0
ΔX_{15}	0	80_x	0	?	?	?	$e_{0,\sim}$	80_x	0	0	0	0
ΔX_{16}	80_x	?	?	?	?	$e_{0,\sim}$	80_x	$e_{0,\sim}$	0	0	0	0
ΔX_{17}	?	?	?	?	$e_{0,\sim}$	?	$e_{0,\sim}$	?	0	0	0	0
ΔX_{17}	?	?	?	$e_{0,\sim}$	80_x	0	?	?	0	0	0	0
ΔX_{18}	?	?	$e_{0,\sim}$	80_x	0	0	?	?	80_x	0	0	0
ΔX_{19}	?	$e_{0,\sim}$	80_x	0	0	0	?	?	0	0	0	0
ΔX_{20}	$e_{0,\sim}$	80_x	0	0	0	0	?	?	0	0	0	0
ΔX_{21}	80_x	0	0	0	0	0	?	$e_{2,\sim}$	0	0	0	0
ΔX_{22}	0	0	0	0	0	0	$e_{2,\sim}$	80_x	0	0	0	0
ΔX_{23}	0	0	0	0	0	0	80_x	0	0	0	0	80_x
ΔX_{24}	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_{25}	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_{26}	0	0	0	0	0	0	0	0	0	0	0	0
ΔX_{27}	0	0	0	0	0	0	0	0	0	0	80_x	0
ΔX_{28}	0	0	0	80_x	0	0	0	0	0	0	0	0

4.3.3.2 The Attack

To conduct 31-round attack, a 22-round related-key impossible differential introduced in the previous subsection is used. This attack recovers all key bytes with the time complexity which

is approximately the same as the exhaustive search. In the attack, firstly the pairs satisfying desired ΔP and ΔC differences respectively are taken, then by doing partial decryption and encryption, the pairs that do not satisfy the impossible differential given in Table 4.3.3.1 are eliminated. Finally, by using remaining pairs, wrong keys are discarded and actual key candidates are kept.

The attack includes rounds from 1 to 31 and final transformation, but excludes initial transformation.

Data Preparation

If rolling back the difference $(0, 0, 0, 0, 0, 0, 80_x, 0)$ through 6 rounds in backward direction, necessary plaintext difference $\Delta P = (?, ?, ?, e_{0,\sim}, 80_x, 0, ?, ?)$ and similarly propagating the difference $(0, 0, 0, 80_x, 0, 0, 0, 0)$ through 3 rounds in forward direction, desired ciphertext differences $\Delta C = (e_{0,\sim}, 80_x, 0, 0, 0, 0, ?, ?)$ are obtained.
Then, the attack algorithm can be expressed as follows:

- Form plaintexts pairs which satisfy the prescribed plaintext difference ΔP = (?, ?, ?,$e_{0,\sim}$, $80_x, 0, ?, ?)$: take a set of 2^{47} plaintexts $P_i's$ which take fixed values in byte 2 and in the right most seven bits of byte 3 and every possible values in bytes (7,6,5,1,0) and in the left most seven bits of byte 4. Then, take another set of plaintexts $P_i^{*\prime}s$ which have 1 in the seventh bit of byte 3 and in the first bit of byte 4 and the remaining bits like as the first set of plaintexts. Also, by changing 15 fixed bits in plaintexts, 2^{15} structures can be constructed. Therefore, $2^{15} \cdot 2^{47} \cdot 2^{47} = 2^{109}$ plaintext pairs are obtained in total. Furthermore, construct another two sets each of which contains 2^{47} plaintexts such that the first set of plaintexts $P_i's$ take 1 in the seventh bit of byte 3 and 0 in the first bit of byte 4 and correspondingly the second set of plaintexts $P_i^{*\prime}s$ take 0 in the seventh bit of byte 3 and 1 in the first bit of byte 4. Here, the number of plaintext pairs is multiplied by two and becomes 2^{110}.

- Encrypt all plaintexts P_i and P_i^* under related keys K and K^* satisfying the difference $K \oplus K^* = (80_x, 0, 0, 0, 0, 0, 0, 0)$ and obtain their corresponding ciphertexts C_i and C_i^*, respectively.

- Take only the ciphertexts pairs C_i and C_i^* such that $C_i \oplus C_i^* = (e_{0,\sim}, 80_x, 0, 0, 0, 0, ?, ?)$ by inserting all ciphertexts into a hash table. This step includes 41-bit filtering condition on ciphertext pairs, so the number of pairs are decreased to 2^{69}.

Plaintext-Ciphertext Pairs Elimination

This part of attack has 23 steps and includes key guesses and elimination of plaintexts-ciphertext pairs which do not satisfy the impossible differential.

1. Guess the original key byte K_0 and deduce the subkey byte SK_0. Since plaintext values are known, calculate the bytes (2,1) of X_1 and take the pairs that satisfy the difference (0,?).
   This step has an 8-bit filtering condition. Therefore, the number of remaining pairs is 2^{61}. The time complexity of this step is $2 \cdot 2^{69} \cdot 2^8 \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{71.05}$.
2. Guess the key byte K_3 and compute the subkey byte SK_3. Then, calculate the bytes (7,0) of X_1.
   The time complexity of this step is $2 \cdot 2^{61} \cdot 2^{16} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{71.05}$.
3. Guess the key byte K_0 and deduce the subkey byte SK_4. Then, calculate the bytes (2,1) of X_2 and take the pairs that satisfy the difference (0,?).
   Here, there is an 8-bit filtering condition, so the number of remaning pairs is 2^{53}.
   The time complexity of this step is $2 \cdot 2^{61} \cdot 2^{24} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{79.05}$.
4. Guess K_9 and compute the whitening key byte WK_4 and SK_{120}, then find the value of bytes (1,0) of X_{30}. Keep the pairs which satisfy the difference (0,?).
   There is an 8-bit filtering condition, so the number of pairs is decreased to 2^{45}.
   The time complexity of this step is $2 \cdot 2^{53} \cdot 2^{32} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{79.05}$.
5. Guess the key byte K_{12} and compute the bytes WK_7 and SK_{123}. Then, find values of bytes (7,6) of X_{30} and take the pairs satisfying the difference $(e_{2,\sim}, 80_x)$.
   In this step, there is a 2-bit filtering condition, so the expected number of remaining pairs is 2^{43}.
   The time complexity of this step is $2 \cdot 2^{45} \cdot 2^{40} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{79.05}$.
6. Compute the value of SK_{119} by using K_0 which is guessed in step 1 and find the values of the bytes (7,6) of X_{29}. Keep the pairs that satisfy the difference $(0, e_{2,\sim})$.
   There is an 8- bit filtering condition which decreases the number of pairs to 2^{35}.

The time complexity of this step is $2 \cdot 2^{43} \cdot 2^{40} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{77.05}$.

7. Guess K_2 and compute the subkey byte SK_2. Then, obtain the bytes (6,5) of X_1.
   The time complexity of this step is $2 \cdot 2^{35} \cdot 2^{48} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{77.05}$.

8. Guess K_7 and calculate SK_7, then obtain the bytes (7,0) of X_2.
   The time complexity of this step is $2 \cdot 2^{35} \cdot 2^{56} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{85.05}$.

9. Guess K_8 and compute the value of SK_8, then find the bytes (2,1) of X_3. Keep the pairs satisfying the difference (0,?). In this step, there is an 8-bit filtering condition which reduces the number of the pairs to 2^{27}.
   The time complexity of this step is $2 \cdot 2^{35} \cdot 2^{64} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{93.05}$.

10. Guess the key byte K_{11} and compute the bytes WK_6 and SK_{122}. Then, find values of bytes (5,4) of X_{30}.
    This step has $2 \cdot 2^{27} \cdot 2^{72} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{93.05}$ time complexity.

11. Compute the value of SK_{118} by using K_{11} which is guessed in the previous step and find the values of the bytes (5,4) of X_{29}.
    The time complexity of this step is $2 \cdot 2^{27} \cdot 2^{72} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{93.05}$.

12. Compute SK_{114} by using previously guessed byte (in step 2) K_3 and find the values of the bytes (5,4) of X_{28}. Then, take the pairs having the difference $(0, 80_x)$.
    This step has 5-bit filtering condition, so the expected number of remaining pairs is 2^{22}. This step has $2 \cdot 2^{27} \cdot 2^{72} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{93.05}$ time complexity.

13. Guess K_1 and compute the subkey byte SK_1. Then, obtain the bytes (4,3) of X_1.
    The time complexity of this step is $2 \cdot 2^{22} \cdot 2^{80} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{96.05}$.

14. Guess K_6 and calculate SK_6, then obtain the bytes (6,5) of X_2.
    The time complexity of this step is $2 \cdot 2^{22} \cdot 2^{88} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{104.05}$.

15. Compute SK_{11} by using previously guessed byte (in step 10) K_{11} and find the values of the bytes (7,0) of X_3.
    This step has $2 \cdot 2^{22} \cdot 2^{88} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{104.05}$ time complexity.

16. Compute SK_{12} by using previously guessed byte (in step 5) K_{12} and find the values of the bytes (2,1) of X_4. Take the pairs with the difference (0,?). There is an 8-bit condition, so the expected number of remaining pairs is 2^{14}.
    This step has $2 \cdot 2^{22} \cdot 2^{88} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{104.05}$ time complexity.

17. Guess K_5 and compute the subkey byte SK_5. Then, obtain the bytes (4,3) of X_2.
    The time complexity of this step is $2 \cdot 2^{14} \cdot 2^{96} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{104.05}$.
18. Guess K_{10} and compute the value of SK_{10}, then find the bytes (6,5) of X_3.
    This step has $2 \cdot 2^{14} \cdot 2^{104} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{112.05}$ time complexity.
19. Guess K_{15} and compute the value of SK_{15}, then find the bytes (7,0) of X_4. Keep the pairs satisfying the difference $(80_x, e_{2,\sim})$. In this step, there is a 2-bit filtering condition which reduces the number of pairs to 2^{12}.
    This step has $2 \cdot 2^{14} \cdot 2^{112} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{120.05}$ time complexity.
20. Compute SK_{16} by using previously guessed byte (in step 8) K_7 and find the values of the bytes (2,1) of X_5. Take the pairs with the difference $(0, e_{2,\sim})$. There is an 8-bit condition, so the expected number of remaining pairs is 2^4.
    The time complexity of this step is $2 \cdot 2^{12} \cdot 2^{112} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{118.05}$.
21. Obtain SK_9 by using previously guessed byte (in step 4) MK_9 and find the values of the bytes (4,3) of X_3.
    $2 \cdot 2^4 \cdot 2^{112} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{110.05}$ is the time complexity of this step.
22. Guess K_{14} and compute the subkey byte SK_{14}. Then, obtain the bytes (6,5) of X_4.
    The time complexity of this step is $2 \cdot 2^4 \cdot 2^{120} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{118.05}$.
23. Compute SK_{19} by using previously guessed byte (in step 7) K_2 and find the values of the bytes (7,0) of X_5.
    This step has $2 \cdot 2^4 \cdot 2^{120} \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{118.05}$.

The total time complexity of all 23 steps is $2^{121.03}$.

Key Elimination Step

Compute SK_{20} by using the byte K_3 which was previously guessed in step 2, then by using SK_{20}, find the values of bytes (2,1) of X_6. Check whether the difference $(0, 80_x)$ is satisfied. If a pair satisfies this difference, conclude that the corresponding guessed key is a wrong key and eliminate it. Since there is a 5-bit filtering condition in this step, it is expected that each plaintext-ciphertext pair eliminates 2^{-5} of 2^{120} keys. Therefore, after the first pair, the number of remaining keys is about $2^{120} - 2^{120} \cdot 2^{-5} = 2^{120}(1 - 2^{-5})$. Then after the second pair, the number of remaining keys is about $2^{120}(1 - 2^{-5}) - 2^{120}(1 - 2^{-5})2^{-5} = 2^{120}(1 - 2^{-5})^2$.

Since there are 2^4 remaining pairs, after 2^4th pair, the expected number of remaining keys is $2^{120}((1-2^{-5})^{2^4}) \approx 2^{119.27}$. The time complexity of this step is $3 \cdot 2^8 \cdot 2 \cdot 2^{120}[1 + (1-2^{-5}) + (1-2^{-5})^2 + \ldots + (1-2^{-5})^{2^4-1}] \cdot \frac{1}{4} \cdot \frac{1}{31} \approx 2^{127.28} \approx 2^{117.89}$. As a final step, for every remaining $2^{119.27}$ key sequences $(K_{15}, K_{14}, K_{12}, \ldots, K_0)$, find key byte K_{13} by making an exhaustive search for 3 plaintext-ciphertext pairs. It is very likely that the actual key is found, because for three pairs, a wrong key is suggested with a probability $(2^{-64})^3 = 2^{-192}$. Hence, the time complexity of the final step is $2^{127.27}$.

So, the total time complexity of the attack is $2^{121.03} + 2^{127.27} \approx 2^{127.28}$.

4.4 Our Related-Key Impossible Differential Distinguisher for XTEA

Tiny Encryption Algorithm (TEA) is a 64-bit block cipher with 128-bit user key which was designed by Needham and Wheeler in 1994 [32]. Afterwards, the designers noticed that TEA has some weaknesses, thus they proposed extended TEA (XTEA) to correct these weaknesses of TEA in 1997 [33]. XTEA has a balanced Feistel structure and is 64-round block cipher with 64-bit block size and 128-bit key size. Like TEA, XTEA is presented to meet the needs of block ciphers having easy implementation property. Therefore, it has a simple structure which has only exclusive-or, addition and shift operations.

Encryption algorithm of XTEA does not satisfy diffusion well. For this reason, differential attack and its extensions have applied to XTEA, so far [34, 14, 35, 31]. In this work, we further analyze XTEA and improve the impossible differential distinguisher for XTEA proposed by Moon et al. [36]. In [36], an impossible differential attack was applied to 14-round XTEA with time complexity of 2^{85} XTEA encryptions and $2^{62.5}$ chosen plaintexts by using 12-round impossible differential. We realize that the key scheduling algorithm of XTEA does not satisfy diffusion well. So, by using differential property of key schedule of XTEA, we increase the number of rounds of the impossible distinguisher from 12 to 25 by using related-keys.

4.4.1 Notations

The notations used in the description of XTEA and the related-key impossible charactresitics are given as follows:

$\oplus$: Exclusive-OR operation

$\boxplus$: Addition in modulo 2^{32}

$\boxtimes$: Multiplication in modulo 2^{32}

($<< i$): i-bit logical shift to the left

($>> i$): i-bit logical shift to the right

K: A 128-bit master key

K_i: 32-bit part of the key K, $1 \leq i \leq 4$

(L_{i-1}, R_{i-1}): Left and right halves of input blocks to the round i, respectively, $1 \leq i < 64$

e_i: A 32-bit having zeros in all positions except the position i

$e_{i,\sim}$: A 32-bit block having zeros in bit positions 0 to $i-1$, in one in ith position, and any values in positions $i+1$ to 31

4.4.2 The XTEA Block Cipher

4.4.2.1 The Key Schedule of XTEA

A 128-bit secret key $K = (K_1, K_2, K_3, K_4)$ is used to generate 64 32-bit round keys k_i's as in the following manner;

$$k_i = K_{(\left\lfloor \frac{i}{2} \right\rfloor \boxtimes \theta >> 11) \ \& \ 3}$$
$$where \theta = 0x9e3779b9$$

The key schedule of XTEA is given in Table 4.4.2.1.

4.4.2.2 The Encryption Function of XTEA

The encryption algorithm of XTEA can be explained as follows:

Table 4.4: The Key Schedule of XTEA

Round i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
k_i	K_1	K_4	K_2	K_3	K_3	K_2	K_4	K_1	K_1	K_1	K_2	K_4	K_3	K_3	K_4	K_2
Round i	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
k_i	K_1	K_1	K_2	K_1	K_3	K_4	K_4	K_3	K_1	K_2	K_2	K_2	K_3	K_1	K_4	K_4
Round i	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
k_i	K_1	K_3	K_2	K_2	K_3	K_2	K_4	K_1	K_1	K_4	K_2	K_3	K_3	K_2	K_4	K_2
Round i	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
k_i	K_1	K_1	K_2	K_4	K_3	K_3	K_4	K_3	K_1	K_2	K_2	K_1	K_3	K_4	K_4	K_3

1. Plaintext $P = (L_0, R_0)$

2. For $i = 1$ to 64;
   $R_i = L_{i-1} \boxplus (((R_{i-1} << 4 \oplus R_{i-1} >> 5) \boxplus R_{i-1}) \oplus (\left\lfloor \frac{i}{2} \right\rfloor \boxtimes \theta \boxplus k_i))$;
   $L_i = R_{i-1}$;

3. Ciphertext $C = (L_{64}, R_{64})$.

The *i*th round of encryption function of XTEA is depicted in figure 4.6.

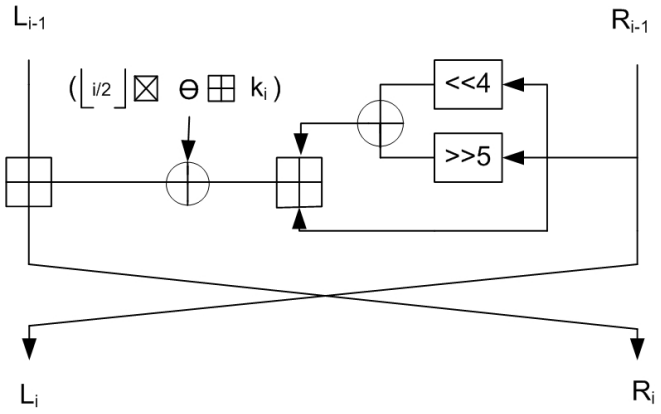

Figure 4.6: *i*th round function of XTEA

4.4.3 A 12-Round Impossible Differential by Moon et al.

In [36], Moon et al. proposed an 12-round impossible distinguisher for XTEA which is given in Table 4.4.3, Then, they built an attack on 14-round of XTEA by adding two rounds at the end of the impossible characteristics.

Table 4.5: A 12-Round Impossible Differential Distinguisher for XTEA

	ΔL_i	ΔR_i
ΔX_i	$e_{25,\sim}$	$a0^{31}$
ΔX_{i+1}	$a0^{31}$	$e_{25,\sim}$
ΔX_{i+2}	$e_{25,\sim}$	$e_{20,\sim}$
ΔX_{i+3}	$e_{20,\sim}$	$e_{15,\sim}$
ΔX_{i+4}	$e_{15,\sim}$	$e_{10,\sim}$
ΔX_{i+5}	$e_{10,\sim}$	$e_{5,\sim}$
ΔX_{i+6}	$e_{5,\sim}$	$e_{0,\sim}$
ΔX_{i+6}	$e_{0,\sim}$	$e_{5,\sim}$
ΔX_{i+7}	$e_{5,\sim}$	$e_{10,\sim}$
ΔX_{i+8}	$e_{10,\sim}$	$e_{15,\sim}$
ΔX_{i+9}	$e_{15,\sim}$	$e_{20,\sim}$
ΔX_{i+10}	$e_{20,\sim}$	$e_{25,\sim}$
ΔX_{i+11}	$e_{25,\sim}$	$b0^{31}$
ΔX_{i+12}	$b0^{31}$	$e_{25,\sim}$

4.4.4 Our 25-Round Related-Key Impossible Distinguisher

Our 25-round related-key impossible differential of XTEA depicted in Table 4.4.4 starts at the beginning of round 19 and ends at the end of round 43;

$(e_{31}, 0) \nrightarrow (0, e_{31})$ under the key difference $(\Delta K_1, \Delta K_2, \Delta K_3, \Delta K_4) = (0, e_{31}, 0, 0)$.

The contradiction occurs at the end of round 32.

Table 4.6: Our 25-Round Related-Key Impossible Differential Characteristic for XTEA

	$\Delta X_{i-1,L}$	$\Delta X_{i-1,R}$	Δk_i
ΔX_{19}	e_{31}	0	e_{31}
ΔX_{20}	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
ΔX_{26}	0	0	e_{31}
ΔX_{27}	0	e_{31}	e_{31}
ΔX_{28}	e_{31}	$e_{26,\sim}$	e_{31}
ΔX_{29}	$e_{26,\sim}$	$e_{21,\sim}$	0
ΔX_{30}	$e_{21,\sim}$	$e_{16,\sim}$	0
ΔX_{31}	$e_{16,\sim}$	$e_{11,\sim}$	0
ΔX_{32}	$e_{11,\sim}$	$e_{6,\sim}$	0
ΔX_{33}	$e_{6,\sim}$	$e_{1,\sim}$	0
ΔX_{33}	$e_{6,\sim}$	$e_{11,\sim}$	0
ΔX_{34}	$e_{11,\sim}$	$e_{16,\sim}$	0
ΔX_{35}	$e_{16,\sim}$	$e_{21,\sim}$	e_{31}
ΔX_{36}	$e_{21,\sim}$	$e_{26,\sim}$	e_{31}
ΔX_{37}	$e_{26,\sim}$	e_{31}	0
ΔX_{38}	e_{31}	0	e_{31}
ΔX_{39}	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
ΔX_{43}	0	0	e_{31}
ΔX_{44}	0	e_{31}	-

The distinguisher given above is constructed by using the following properties of XTEA:

Property 1: *The key schedule of XTEA does not satisfy diffusion well: if a difference is given to the only one part of the key, then about 75% of all subkeys has zero difference.*

Property 2: *If the difference is in the most significant bit position, then modular addition operation* ($\boxplus$) *behaves like bitwise Exclusive OR operation* ($\oplus$).

Our 25-round related-key impossible distinguisher is constructed by giving difference to only one part of the master key K in order to let the difference propagates slowly. The construction of the distinguisher can briefly discussed as follows:

- Begin by giving the input difference $(e_{31}, 0)$ to round 19 which is cancelled by the subkey difference e_{31} of round 19 by the Property 1. Thus, the output difference of round 19 becomes zero.

- This zero difference is preserved through rounds 20,21,22,23,24,25 until the end of round 26 due to the nonzero key difference in round 26. Therefore, the output difference of round 26 will be $(0, e_{31})$.
  Note that the reason of that the characteristics begins at round 19 and has difference in K_2 is to keep zero difference as many rounds as possible.

- Then, the difference $(0, e_{31})$ propagates the following 6 rounds and at the end of round 32 the output difference becomes $(e_{6,\sim}, e_{1,\sim})$. This completes our first 14-round differential.

- Now, at the end of round 43, if the output difference is chosen as $(0, e_{31})$ and is rolled backward, then this difference is cancelled by the nonzero subkey difference e_{31} of round 43.

- This zero difference is kept until round 38, because of the nonzero subkey difference of round 38. Thus, the output difference of round 37 will be $(e_{31}, 0)$.

- If the difference $(e_{31}, 0)$ is rolled back through 5 rounds, then the output difference of round 32 becomes $(e_{6,\sim}, e_{11,\sim})$. This completes our second 11-round differential.
- If two differentials are concatenated, it is seen that the output difference $(e_{6,\sim}, e_{1,\sim})$ of round 32 of the first differential can never be equal to the output difference $(e_{6,\sim}, e_{11,\sim})$ of round 32 of the second differential that gives rise to a contradiction at this round.

To conclude, if the differential property of key schedule of XTEA is taken in to consideration, a 12-round impossible distinguisher can be improved.

CHAPTER 5

RELATED-KEY BOOMERANG ATTACK AND ITS EXTENSIONS

This chapter is mainly devoted to related-key boomerang attack and its extensions: related-key amplified boomerang, related-key rectangle and related-key impossible boomerang attacks. Throughout this chapter, Section 5.1 gives description of boomerang attack and exemplifies the attack on reduced round IDEA proposed by Biham et al. [37]. Sections 5.2 and 5.3 describes related-key amplified boomerang and related-key rectangle attacks, respectively. Section 5.3 also includes an attack on full round SHACAL-1. Finally, Section 5.4 discusses a new attack called related-key impossible boomerang attack.

5.1 Related-Key Boomerang Attack

The Boomerang attack was proposed by D. Wagner in 1999 and firstly applied to the block cipher COCONUT98 [38]. The Boomerang attack is based on differential cryptanalysis and uses two short differential with high probability instead of using one long low probability differential. Afterwards, Kim et al. have combined the boomerang attack with the related-key idea which they call related-key boomerang attack in [39] and applied the attack to IDEA and COCONUT98. In [39], a boomerang attack which uses one ordinary differential and one related-key differential is proposed. Then, in [40] and [37], a boomerang attack using two related-key differentials is presented, independently.

Notice that in order to perform related-key boomerang attack, the attacker should have a power to make encryption and decryption under unknown related-keys. To describe the related-key

boomerang attack, consider the block cipher $E : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ as a cascade of two subciphers E_0 and E_1, i.e. $E = E_1 \circ E_0$ such that for E_0 there is a differential $\alpha \rightarrow \beta$ with probability p and for E_1 there is a differential $\gamma \rightarrow \delta$ with probability q. Note that, if the differentials $\alpha \rightarrow \beta$ for E_0 and $\gamma \rightarrow \delta$ for E_1 hold with probability p and q, respectively, then the differentials $\beta \rightarrow \alpha$ for E_0^{-1} and $\delta \rightarrow \gamma$ for E_1^{-1} also hold with probability p and q, respectively. The important point of the related-key boomerang attack is to combine these two differentials. Then, by using these two related-key differentials, related-key boomerang distinguisher based on four related keys can be constructed as follows:

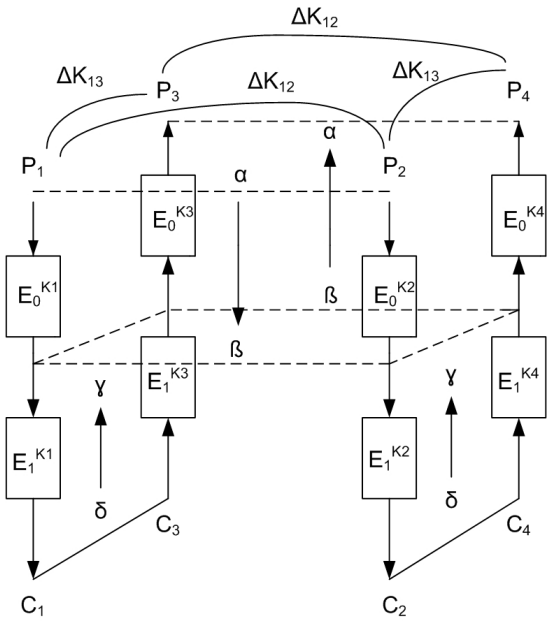

Figure 5.1: Related-key boomerang distinguisher based on four related keys

- Take a randomly chosen plaintext P_1 and form another plaintext P_2 such that $P_2 = P_1 \oplus \alpha$.
- Obtain their corresponding ciphertexts $C_1 = E_{K_1}(P_1)$ and $C_2 = E_{K_2}(P_2)$ where $K_2 = K_1 \oplus \Delta K_{12}$.
- Form another pair of ciphertexts C_3 and C_4 such that $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$.
- Obtain their corresponding plaintexts $P_3 = E_{K_3}^{-1}(C_3)$ and $P_4 = E_{K_4}^{-1}(C_4)$ where $K_3 = K_1 \oplus \Delta K_{13}$ and $K_4 = K_1 \oplus \Delta K_{12} \oplus \Delta K_{13}$.

- Check $P_3 \oplus P_4 = \alpha$.

To analyze the steps of the algorithm, we assume that the differential $\alpha \rightarrow \beta$ for E_0 holds with probability p in the second step. Then, in the fourth step, we expect that the differential $\delta \rightarrow \gamma$ for E_1^{-1} holds with probability q for both ciphertext pairs (C_1, C_3) and (C_2, C_4). Once $(E_1^{K_3})^{-1}(C_3) \oplus (E_1^{K_4})^{-1}(C_4) = \beta$ is automatically satisfied, the difference $(E_0^{K_3})^{-1}((E_1^{K_3})^{-1}(C_3)) \oplus (E_0^{K_4})^{-1}((E_1^{K_4})^{-1}(C_4)) = P_3 \oplus P_4 = \alpha$ holds with probability p. The intermediate difference $(E_1^{K_3})^{-1}(C_3) \oplus (E_1^{K_4})^{-1}(C_4) = \beta$ is always satisfied due to the following equation:

$$
\begin{aligned}
&(E_1^{K_3})^{-1}(C_3) \oplus (E_1^{K_4})^{-1}(C_4) \\
&= (E_1^{K_3})^{-1}(C_3) \oplus (E_1^{K_4})^{-1}(C_4) \oplus (E_1^{K_1})^{-1}(C_1) \oplus (E_1^{K_1{}^{-1}})(C_1) \oplus (E_1^{K_2})^{-1}(C_2) \oplus (E_1^{K_2})^{-1}(C_2) \\
&= (E_1^{K_1})^{-1}(C_1) \oplus (E_1^{K_3})^{-1}(C_3) \oplus (E_1^{K_2})^{-1}(C_2) \oplus (E_1^{K_4})^{-1}(C_4) \oplus (E_1^{K_1}) {-} 1(C_1) \oplus (E_1^{K_2})^{-1}(C_2) \\
&= \gamma \oplus \gamma \oplus (E_1^{K_1})^{-1}(C_1) \oplus (E_1^{K_2})^{-1}(C_2) = E_0^{K_1}(P_1) \oplus E_0^{K_2}(P_2) \\
&= \beta.
\end{aligned}
$$

A right plaintext quartet (P_1, P_2, P_3, P_4) is the one which satisfies all four differentials simultaneously.

Therefore, the probability of related-key boomerang distinguisher depicted in Figure 5.1 is $p^2 \cdot q^2$. Notice that for a random n-bit permutation last step holds with probability 2^{-n}. For this reason, the related-key boomerang distinguisher works only if $p^2 \cdot q^2 > 2^{-n}$ or equivalently $p \cdot q > 2^{\frac{-n}{2}}$.

5.1.1 Related-Key Boomerang Attack on Reduced-Round IDEA

The related-key boomerang attack is applied to six 6-round IDEA by using 5.5-round related-key boomerang distinguisher [37]. The attack captures 32 bits of the secret key K by using four related-keys with $2^{51.6}$ and 2^{48} data and time complexities, respectively. Now, we begin by giving the notations used in the attack.

5.1.1.1 Notations

The notations used in the description of IDEA and the attack are given as follows:

$\boxplus$: Addition in modulo 2^{16}

$\odot$: Multiplication in modulo $2^{16} + 1$

X_j^i: *jth* part of the input to round i, $1 \leq i \leq 8$, $1 \leq j \leq 4$

Z_j^i: *jth* subkey of round i, $1 \leq i \leq 8$, $1 \leq j \leq 6$

Y_j^i: *jth* part of the output of Key Mixing of round i, $1 \leq i \leq 8$, $1 \leq j \leq 4$

5.1.1.2 The IDEA Block Cipher

The block cipher IDEA (International Data Encryption Algorithm) was proposed by Lai, James and Massey in 1990 [41]. It is 8.5-round with 64-bit block size and 128-bit key size. IDEA uses two operations in its encryption function defined as follows:

Key mixing(T) : This operation which is denoted by T divides 64-bit input value into four equal 16-bit words, then mixes subkeys with these 16-bit words by using modular addition ($\boxplus$) in modulo 2^{16} and multiplication ($\odot$) in modulo $2^{16} + 1$.

M mixing (M) : This operation includes a mutiplication-addition structure denoted by MA and a swap of two middle words denoted by s and can be written as $M = s \circ MA$.

Note that 8.5 round IDEA can be expressed as $T \circ s \circ (M \circ T)^8$ and one round of IDEA is depicted in figure 5.2.

Key schedule of IDEA

The Key Schedule of IDEA transforms 128-bit secret key into fifty two 16-bit subkey words. Each round uses 6 subkey words such that 4 of them are used in the key mixing operation and

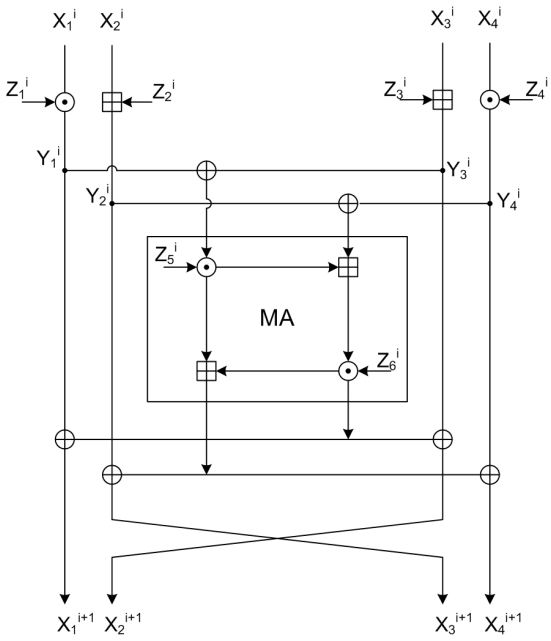

Figure 5.2: *i*th round of IDEA

2 of them are used in the M mixing operation.

The key schedule of IDEA is given in Table 5.1.

Table 5.1: Key Schedule of IDEA

Round(i)	$Z_1{}^i$	$Z_2{}^i$	$Z_3{}^i$	$Z_4{}^i$	$Z_5{}^i$	$Z_6{}^i$
1	0-15	16-31	32-47	48-63	64-79	80-95
2	96-111	112-127	25-40	41-56	57-72	73-88
3	89-104	105-120	121-8	9-24	50-65	66-81
4	82-97	98-113	114-1	2-17	18-33	34-49
5	75-90	91-106	107-122	123-10	11-26	27-42
6	43-58	59-74	100-115	116-3	4-19	20-35
7	36-51	52-67	68-83	84-99	125-12	13-28
8	29-44	45-60	61-76	77-92	93-108	19-124
Last half round	22-37	38-53	54-69	70-85	-	-

5.1.1.3 A Related-Key Boomerang Attack 6-Round IDEA

A 5.5-Round Related-Key Boomerang Distinguisher for IDEA

The 5.5-round distinguisher starts at the beginning of round 2 and ends at the end of mixing operation in round 6 and can be considered as $T \circ (M \circ T)^5$. It is built by combining two differentials which are depicted in figure 5.3. The first differential $(M \circ T)^3$ has 3 rounds and the second differential $T \circ (M \circ T)^2$ has 2.5 rounds.

The related-key differentials visualized in Figure 5.3 can be constructed as follows:

The First Differential in the forward direction: If the key difference is chosen as $\Delta K = e_{25}$ and the input difference to round 2 is chosen as $\alpha = (0, 0, e_{15}, 0)$, then, the nonzero difference α is cancelled by the subkey difference $\Delta Z_3^2 = e_{15}$ and the zero difference is preserved until the MA of in round 4. The subkey difference $\Delta Z_5^4 = e_8$ makes the output difference of MA an unknown difference $\beta = (\beta_1, \beta_2)$ where β_1 and β_2 has 16-bit halves of β. This unknown β difference has at most 2^{32} different values because the output of the MA is 32-bit word. Since, it is assumed that all β's have the equal probability which is 2^{-32}, the probability of this differential can be computed as: $\hat{p} = \sqrt{\sum_{2^{32}}(2^{-32})^2} = 2^{-16}$.

The Second Differential in the backward direction: The key difference is chosen as $\Delta K' = e_{75}$ and the output difference of the key mixing operation M in round 6 is chosen as $\delta = (0, 0, e_8, 0)$. If the difference is rolled back through the M, then it is cancelled by the subkey difference $\Delta Z_3^7 = e_8$ with probability $\frac{1}{2}$. The zero difference is preserved till the beginning of the M in round 5. The key subkey difference $\Delta Z_1^5 = e_{15}$ leads to the input of round 5 unknown 16-bit γ difference which has at most 2^{16} different values. In addition, it is assumed that all γ's have the equal probability which is 2^{-17}(This is computed experimentally by the attackers, the probabilities are not equiprobable and $\hat{q} > 2^{-8.8}$). Therefore, the probability of this differential can be computed as: $\hat{q} = \sqrt{\sum_{2^{16}}(2^{-17})^2} = 2^{-18} = 2^{-9}$.

The Attack

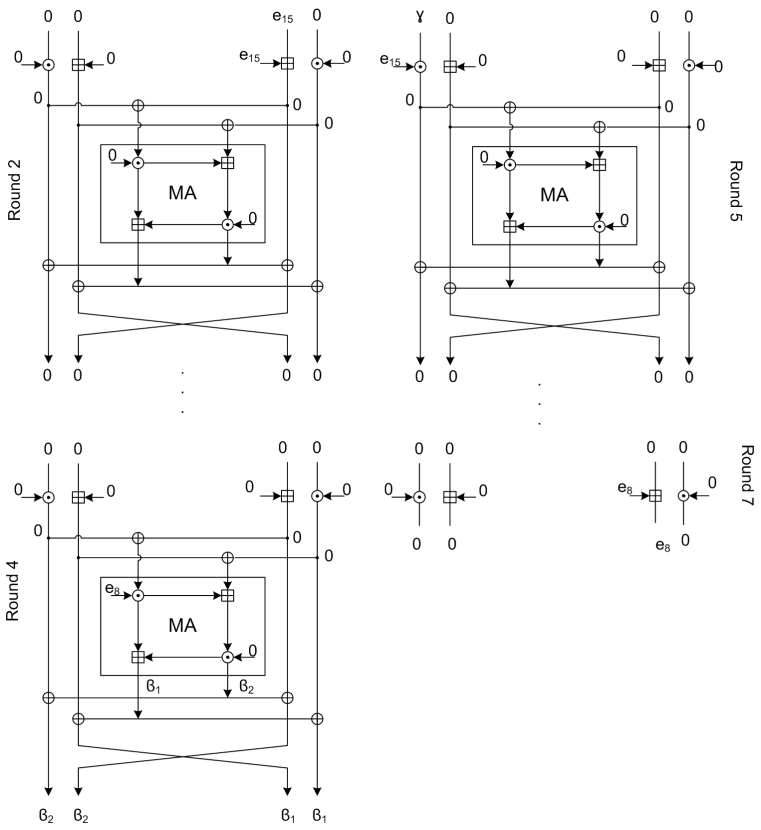

Figure 5.3: The first related-key differential is on the left-hand side and the second related-key differential is on the right-hand side for IDEA

The attack which utilizes 5.5 round related-key boomerang distinguisher given in the previous subsection starts at the MA in round 1 and ends just before the MA in round 7. The attack captures the bits in the position [64-95] of the original key K.

As in the construction of the distinguisher, in the attack, the difference between the original key and related-keys are choosen as $K_2 = K_1 \oplus e_{25}$, $K_3 = K_1 \oplus e_{75}$ and $K_4 = K_1 \oplus e_{25} \oplus e_{75}$. Then, the attack works as follows:

- Start a counter initialized to zero, for each guess of [64-95] of K_1
- Construct a set of $2^{17.6}$ 32-bit value (e, f) such that for each 32-bit value:
    1. Form a set of 2^{32} plaintexts $\mathcal{P}_1$ such that each plaintext in $\mathcal{P}_1$ has the form $P = (a, b, c, d)$ and $a \oplus c = e$ and $b \oplus d = f$.
    2. Form another set of 2^{32} plaintexts $\mathcal{P}_2$ such that each plaintext in $\mathcal{P}_2$ has the form $P_2 = (a, b \oplus e_{15}, c, d)$ and $a \oplus c = e$ and $b \oplus d = f$.
    3. Encrypt the set $\mathcal{P}_1$ under K_1 and obtain the set of corresponding ciphertexts C_1.
    4. Encrypt the set $\mathcal{P}_2$ under K_2 and obtain the set of corresponding ciphertexts C_2.
    5. Construct a set of ciphertexts C_3 such that each ciphertext in C_3 has the form $C_3 = C_1 \oplus \delta$, where $\delta = (0, 0, e_{15}, 0)$.
    6. Construct another set of ciphertexts C_4 such that each ciphertext in C_4 has the form $C_4 = C_2 \oplus \delta$, where $\delta = (0, 0, e_{15}, 0)$.
    7. Decrypt the sets C_3 and C_4 under K_3 and K_4, respectively and obtain their sets of corresponding plaintexts $\mathcal{P}_3$ and $\mathcal{P}_4$.
    8. Build a hash table storing the plaintexts in $\mathcal{P}_3$ and $\mathcal{P}_4$ indexed by the XOR value of the first and third 16-bit words and by the XOR value of the second and fourth 16-bit words.
    9. For any guess of K_1 in the positions [64-95]:
        - Consider a pair of colliding plaintexts (P_3, P_4) and partially encrypt the plaintexts P_1, P_2, P_3 and P_2 through the MA in round 1; compute the difference between the partially encrypted values of P_1 and P_2 and the difference between the partially encrypted values of P_3 and P_4, if these are both equal to

the difference α, then go to the next step. If this is not the case, try the next subkey value.

- Partially decrypt the respective ciphertext pairs. Then compute the difference of partially decrypted values of ciphertext pairs. If the difference is zero, increment the counter of the subkey by one. (This step should be done, because the guessed key value [64-95] of K_1 includes the subkey $Z_3^7 = e_{15}$).

- Conclude that the right subkey is the one which has the highest counter.

The attack needs $2^{51.6}$ plaintext/ciphertexts such that $2^{49.6}$ plaintexts are encrypted under K_1, $2^{49.6}$ plaintexts are encrypted under K_2, $2^{49.6}$ ciphertexts are decrypted under K_3 and $2^{49.6}$ ciphertexts are decrypted under K_4. The time complexity of the attack is $2^{51.6}$ MA encryptions or equivalently 2^{48} 6-round IDEA encryptions.

5.2 Related-Key Amplified Boomerang Attack

Amplified boomerang attack is the refinement of boomerang attack that transforms the adaptively chosen ciphertext nature of the boomerang attack into chosen plaintext attack with the cost of having lower probability. Namely, contrary to boomerang attack, the attacker only needs to make encryptions under secret key.

Afterwards, amplified boomerang attack is combined with the related-key idea which is called related-key amplified boomerang attack. This attack was applied to the block ciphers Eagle-64 and Eagle-128 by Jeong et al. [42] in 2007.

The related-key amplified boomerang attack tries to capture a set of quartets of plaintexts satisfying the difference $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$. Each pair satisfies the differential $\alpha \rightarrow \beta$ of E_0 with probability p. If $E_0(P_1)$ and $E_0(P_3)$ satisfy the difference $E_0(P_1) \oplus E_0(P_3) = \gamma$ with probability of 2^{-n}, then the difference $E_0(P_2) \oplus E_0(P_4) = \gamma$ holds for free due to the boomerang conditions mentioned in the boomerang attack. Afterwards, for the differential $\gamma \rightarrow \delta$ of E_1, two pairs satisfy the differences $C_1 \oplus C_3 = \delta$, $C_2 \oplus C_4 = \delta$ with probability of $2^{-n} \cdot P^2 \cdot q^2$. The construction of the related-key amplified boomerang distinguisher depicted

in figure 5.5 based on four related-keys can be given as follows:

- Take a randomly chosen plaintext P_1 and form a plaintext P_2 such that $P_1 \oplus P_2 = \alpha$.
- Obtain their corresponding ciphertexts $C_1 = E_{K_1}(P_1)$ and $C_2 = E_{K_2}(P_2)$ where $K_2 = K_1 \oplus \Delta K_{12}$.
- Take another randomly chosen plaintext P_3 and form a plaintext P_4 such that $P_3 \oplus P_4 = \alpha$.
- Obtain their corresponding ciphertexts $C_3 = E_{K_3}(P_3)$ and $C_4 = E_{K_4}(P_4)$ where $K_3 = K_1 \oplus \Delta K_{13}$ and $K_4 = K_1 \oplus \Delta K_{12} \oplus \Delta K_{13}$.
- Check $C_1 \oplus C_3 = \delta$ and $C_2 \oplus C_4 = \delta$.

A quartet which satisfies all prescribed differentials is called *a right quartet*. If we start taking m_1 pairs of plaintexts (P_1, P_2) and m_2 pairs of plaintexts (P_3, P_4), then the expected number of quartets satisfying differential $\alpha \rightarrow \beta$ is $m_1 \cdot m_2 \cdot p^2$. Therefore, it is expected to have $2^{-n} \cdot m_1 \cdot m_2 \cdot p^2 \cdot q^2$ right quartets. Since for any random permutation, about $m_1 \cdot m_2 \cdot 2^{-n}$ right quartets are expected, the distinguisher works only if $p \cdot q > 2^{-n/2}$ and can be used to distinguish the cipher from a random permutation [42].

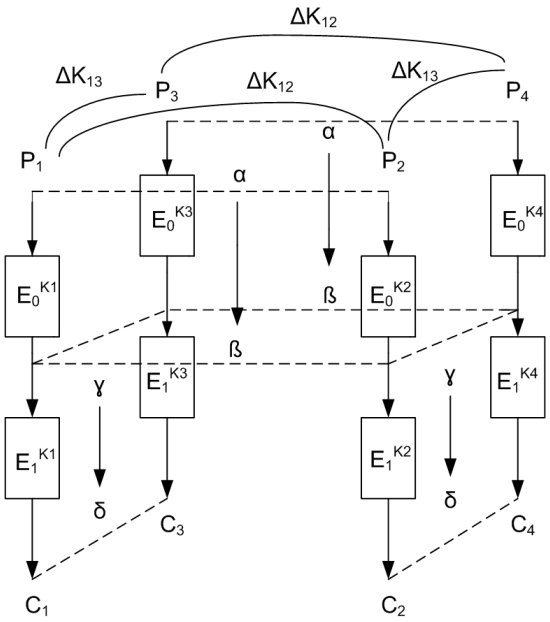

Figure 5.4: Related-key amplified boomerang distinguisher based on four related keys

In the following section, the related-key rectangle attack which is a refinement of the related-key amplified boomerang attack is mentioned.

5.3 Related-Key Rectangle Attack

Rectangle attack [43] is based on differential cryptanalysis and converts the feature of adaptively chosen attack of boomerang attack into chosen plaintext attack. Then, rectangle attack is combined with the related key idea by J. Kim et al. [39] and applied to many known block ciphers [39, 37, 40, 44, 45]. Like related-key boomerang attack, related-key rectangle attack takes advantage of differential properties of the encryption algorithm and key schedule of the cipher and uses shorter related-key differentials. The main difference between related-boomerang attack and related-key rectangle attack is that related-key rectangle attack uses two differentials E_0 and E_1 in only encryption direction, while related-key boomerang attack uses two differentials E_0 and E_1 in both encryption/decryption directions. This property of rectangle attack makes it more realistic to be implemented.

Related-key rectangle attack can be based on 2,4 or more related-keys. As in the boomerang attack, the attack treats the cipher E as a cascade of two sub-ciphers E_0 and E_1, that is $E = E_1 \circ E_0$. Assume that we have a related-key differential $\alpha \rightarrow \beta$ of E_0 with probability p and a related-key differential $\gamma \rightarrow \delta$ of E_1 with probability q. Using two related-key differentials, the related-key rectangle distinguisher depicted in figure 5.5 based on four related keys can be constructed as follows:

- Take a randomly chosen plaintext P_1 and form a plaintext P_2 such that $P_1 \oplus P_2 = \alpha$.
- Obtain their corresponding ciphertexts $C_1 = E_{K_1}(P_1)$ and $C_2 = E_{K_2}(P_2)$ where $K_2 = K_1 \oplus \Delta K_{12}$.
- Take another randomly chosen plaintext P_3 and form a plaintext P_4 such that $P_3 \oplus P_4 = \alpha$.
- Obtain their corresponding ciphertexts $C_3 = E_{K_3}(P_3)$ and $C_4 = E_{K_4}(P_4)$ where $K_3 = K_1 \oplus \Delta K_{13}$ and $K_4 = K_1 \oplus \Delta K_{12} \oplus \Delta K_{13}$.
- Check $C_1 \oplus C_3 = \delta$ and $C_2 \oplus C_4 = \delta$.

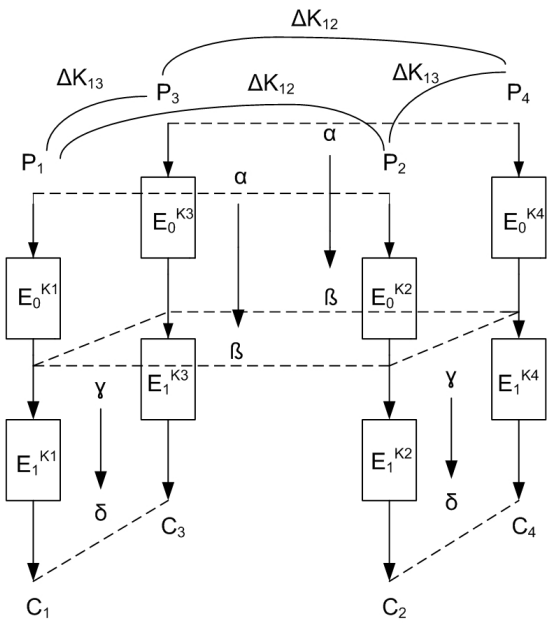

Figure 5.5: Related-key rectangle distinguisher based on four related keys

Related-key rectangle attack is the improvement version of related-key amplified boomerang attack, because related-key rectangle attack takes advantage of all β''s and γ''s satisfying $\alpha \rightarrow \beta'$ and $\gamma' \rightarrow \delta$. A *right quartet* can be defined as a quartet of plaintexts (P_1, P_2) and (P_3, P_4) and their corresponding ciphertext that satisfies the following equalities:

$$P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$$

$$C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$$

As the second improvement, the pair (P_1, P_2) and (P_3, P_4) composes two different quartets $((P_1, P_2), (P_3, P_4))$ and $((P_1, P_2), (P_4, P_3))$ which reduces the data complexity of the attack. The probability P of related-key rectangle distinguisher is $2^{-n}(\hat{p} \cdot \hat{q})^2$ where $\hat{p} = \sqrt{\sum_\beta Pr^2_{K_1,K_2}[\alpha \rightarrow \beta]}$ and $\hat{q} = \sqrt{\sum_\gamma Pr^2_{K_3,K_4}[\gamma \rightarrow \delta]}$.

For a random permutation, the probability of any quartet which satisfies the boomerang distinguisher is $2^{-2n}K$ where K the cardinality of the set of all possible δ's. Therefore, if $P > 2^{-2n}K$ is satisfied, then the related-key rectangle distinguisher works. It can be conclude that if we take N plaintext pairs, it is expected to have $N^2 \cdot 2^{-n}(\hat{p} \cdot \hat{q})^2$ right quartets.

5.3.1 A Related-Key Rectangle Attack on the Full Round SHACAL-1

SHACAL-1 is a block cipher proposed by H. Handschuh and D. Naccache to the NESSIE project in 2000 [12]. Afterwards, many known attacks have been applied to SHACAL-1, however, the most efficient ones are the related-key attacks which enable to break full-round SHACAL-1 [46, 11]. A rectangle attack on 49-round SHACAL-1 was proposed by E. Biham with data complexity of $2^{151.9}$ chosen plaintexts and time complexity of $2^{508.5}$ 49-round SHACAL-1 encryptions [45]. Then, In [39], the attack has been further analyzed by considering key differences and a related-key rectangle attack have been applied to 59-round SHACAL-1. This attack needs $2^{149.7}$ related-key chosen plaintexts and has time complexity of $2^{498.3}$ 59-round SHACAL-1 encryptions. Then, this attack has been improved by increasing the number of attacking rounds from 59 to 70 [40] by using $2^{151.8}$ related-key chosen plaintexts and a running time of $2^{500.1}$ 70-round SHACAL-1 encryptions. Finally, Dunkelman et al. [46] have proposed a related-key rectangle attack on 80-round (full) SHACAL-1 which needs $2^{159.8}$ related-key chosen plaintexts and has time complexity of 2^{423} 80-round SHACAL-1 encryptions.

In this section, we will mention the best related-key rectangle attack on SHACAL-1 [46] which captures 352 bits of the secret key K.

5.3.1.1 Notations

The notations used in the description of SHACAL-1 and the attack are given as follows:
e_i: A 32-bit word having zeros in all positions except the position i.
$e_{i,j}$: A 32-bit word having zeros in all positions except the positions i and j, that is $e_i \oplus e_j$.
$\{A_i, B_i, C_i, D_i, E_i\}$: The input to the round i.
$X^{j-1,i}$: ith part of the input to the round j, $i \in \{A, B, C, D, E\}$.

5.3.1.2 The SHACAL-1 Block Cipher

SHACAL-1 is a 160-bit block cipher which uses variable length secret key(0-512) and based on the compression function of SHA-1 [48]. It has an iterative 80-round structure which is

feistel-type with five branches.

Key Schedule of SHACAL-1

A 512-bit secret key $K = (K_0, K_1, \ldots K_{15})$ is used to generate eighty 32-bit subkeys $k_0, k_1, \ldots k_{127}$ in the following manner:

$$k_i = \begin{cases} K_i, & 0 \leq i \leq 15 \\ (k_{i-3} \oplus k_{i-8} \oplus k_{i-14} \oplus k_{i-16}), & 16 \leq i \leq 79 \end{cases}$$

Encryption Algorithm

Encryption function of SHACAL-1 splits the 160-bit plaintext block P into five 32-bit words A_0, B_0, C_0, D_0 and E_0, then applies the following algorithm:

Input: Plaintext= $(A_0, B_0, C_0, D_0, E_0)$

For $i = 0$ to $i = 79$;

$A_{i+1} = k_i + ROL_5(A_i) + f_i(B_i, C_i, D_i) + E_i + \delta_i$

$B_{i+1} = A_i$

$C_{i+1} = ROL_{30}(B_i)$

$D_{i+1} = C_i$

$E_{i+1} = D_i$

Output: Ciphertext=$(A_{80}, B_{80}, C_{80}, D_{80}, E_{80})$

Where δ_i is the round constant.

Furthermore, encryption function of SHACAL-1 uses three different functions f_i each of which are used in different rounds of the encryption process:

$$f_i = \begin{cases} f_{if} = (X \ \& \ Y) \,|\, (\neg X \ \& \ Z), & 0 \leq i \leq 19 \\ f_{xor} = (X \oplus Y \oplus Z), & 19 \leq i \leq 39, 60 \leq i \leq 79 \\ f_{maj} = (X \ \& \ Y) \,|\, (X \ \& \ Z) \,|\, (Y \ \& \ Z) & 40 \leq i \leq 59 \end{cases}$$

Notice that ith round of the encryption function of SHACAL-1 is depicted in Figure 5.6.

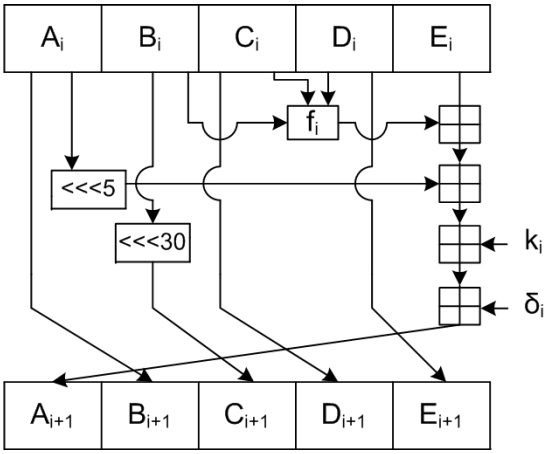

Figure 5.6: *i*th round function of SHACAL-1

5.3.1.3 Related-Key Rectangle Attack on Full Round SHACAL-1

The related-key rectangle attack presented in [46] is built on 69-round related-key rectangle distinguisher. The attack captures $11 \cdot 32 = 352$ bits of the user key K. Firstly, we begin by giving the distinguisher for SHACAL-1 and then explain the attack in detail.

5.3.1.4 A 69-Round Related-Key Rectangle Distinguisher for SHACAL-1

The differentials used in the attack is based on the differentials proposed in [47]. SHACAL-1 is decomposed into two subciphers E_0 and E_1 such that the first differential E_0 given in Table 5.2 includes rounds 0-33 with probability of $p = 2^{-41}$ and the second differential E_1 depicted in Table 5.3 includes rounds 34-68 with probability of $p = 2^{-39}$.

Related-key rectangle distinguisher can be built by counting over possible differentials. Therefore, for the first differential, if the most significant bit of A of the plaintext is fixed to 0, then the probability of round 2 will increase a factor of 2, i.e $p = 2^{-1}$. Similarly, if the third bit of A is set to differ from the third bit of B, then the probability of round 3 increase by a factor of 2, i.e $p = 1$. Consequently, by using these differentials which have the same 33-round differential path as in Table 5.2, the probability of the first differential increases to $\hat{p} = 2^{38.5}$. Similarly, for the second differential, by using all possible γ' differences, the probability of the second differential will rise to $\hat{q} = 2^{38.3}$. Therefore, the probability P of the distinguisher

is computed as $P = 2^{-160} \cdot (2^{-38.5} \cdot 2^{-38.3})^2 = 2^{-313.6}$.

5.3.1.5 The Attack

The attack can be performed by using 69-round related-key rectangle distinguisher given in the previous subsection. To summarize the attack procedure, firstly plaintext quartets are formed, then, their corresponding ciphertexts are decrypted through last 11 rounds by guessing the subkeys of these rounds, and finally, the distinguisher is applied to identify the actual key bits.

Then, the data preparation and the attack algorithm are given as follows:

Data Preparation

- Form $2^{157.8}$ pairs of plaintexts (P_1, P_2) such that $P_1 \oplus P_2 = \alpha$, where $\alpha = (0, 0, e_{31}, e_{31}, e_{31})$ and obtain their corresponding ciphertexts (C_1, C_2) under related-keys K_1 and K_2, respectively.
- Form another $2^{157.8}$ pairs of plaintexts (P_3, P_4) such that $P_3 \oplus P_4 = \alpha$, where $\alpha = (0, 0, e_{31}, e_{31}, e_{31})$ and obtain their corresponding ciphertexts (C_3, C_4) under secret keys K_3 and K_4, respectively.

As computed in the previous subsection, the probability of the distinguisher is $P = 2^{-313.6}$. Since $2 \cdot 2^{157.8}$ plaintext pairs are taken, there are $(2^{157.8})^2 = 2^{315.6}$ quartets of which $2^{315.6} \cdot 2^{-313.6} = 2^2 = 4$ are expected to be *right quartets*. Then, the attack algorithm can be expressed as follows:

1. Guess the subkeys of rounds $72 - 79$;
   - Partially decrypt all ciphertexts under respective through rounds $72 - 79$.
   - Detect all partially decrypted values of pairs (X_1^{71}, X_3^{71}) which satisfy $(X_1^{71,A,B,C} \oplus X_3^{71,A,B,C}) \in S$ and S is defined as $S = (x, y, z) : ROL_{30}(x) \in S', ROL_{30}(y) = 0, ROTL_{30}(z) = e_2$, where S' is the set of possible differences of $X^{70,A}$.
   - Keep only the quartets (P_1, P_2, P_3, P_4) such that $(X_2^{71,A,B,C} \oplus X_4^{71,A,B,C}) \in S$ and go to the next step.

This step has time complexity of $\frac{8}{80} \cdot (2^{32})^8 \cdot 2 \cdot 2^{157.8} = 2^{411.5}$ SHACAL-1 encryptions. Since the set S' has only $2^{8.3} = 224$ values, this step has $2 \cdot (64 + 23.7) = 175.4$-bit filtering condition. Therefore, the expected number of surviving quartets is $2^{315.6} \cdot 2^{-175.4} = 2^{140.2}$.

2. Guess the subkey of round 71;

   - Partially decrypt all surviving quartets under respective subkeys and obtain the value $(X_1^{70}, X_2^{70}, X_3^{70}, X_4^{70})$.
   - For each of surviving quartets, compute the value $(X_1^{70,E} \oplus X_3^{70,E})$ and keep only the quartets such that $(X_1^{70,E} \oplus X_3^{70,E} = 0)$.
   - Then, for each of surviving quartets, compute the value $(X_2^{70,E} \oplus X_4^{70,E})$ and keep only the quartets such that $(X_2^{70,E} \oplus X_4^{70,E} = 0)$ and go to the next step.
     The time complexity of this step has $\frac{1}{80} \cdot (2^{32})^8 \cdot 2^{32} \cdot 2 \cdot 2^{140.2} = 2^{422.9}$. This step has $(32 + 32) = 64$-bit filtering condition, therefore there are about $2^{140.2} \cdot 2^{-64} = 2^{76.2}$ quartets left.

3. Guess the subkey of round 70;

   - Partially decrypt all surviving quartets under respective keys and obtain the value $(X_1^{69}, X_2^{69}, X_3^{69}, X_4^{69})$.
   - For each of surviving quartets, compute the value $(X_1^{69,E} \oplus X_3^{69,E})$ and keep only the quartets such that $(X_1^{69,E} \oplus X_3^{69,E} = 0)$.
   - Then, for each of surviving quartets, compute value $(X_2^{69,E} \oplus X_4^{69,E})$ and keep only the quartets such that $(X_2^{69,E} \oplus X_4^{69,E} = 0)$ and go to the next step.
     The time complexity of this step has $\frac{1}{80} \cdot (2^{32})^8 \cdot 2^{32} \cdot 2^{32} \cdot 2 \cdot 2^{75.2} = 2^{390.9}$. This step has $(32 + 32) = 64$ bits filtering condition, hence, about $2^{140.2} \cdot 2^{-64} = 2^{76.2}$ quartets are expected to survive.

4. Guess the subkey of round 69;

   - Partially decrypt all remaining quartets under respective keys and obtain the value $(X_1^{68}, X_2^{68}, X_3^{68}, X_4^{68})$.
   - For each of surviving quartets, compute the value $(X_1^{68,E} \oplus X_3^{68,E})$ and keep only the quartets such that $(X_1^{68,E} \oplus X_3^{68,E} = e_1)$.

- Then, for each of remaining quartets, compute the value $(X_2^{68,E} \oplus X_4^{68,E})$ and keep only the quartets such that $(X_2^{68,E} \oplus X_4^{68,E} = e_1)$ and go to the next step.

  The time complexity of this step has $\frac{1}{80} \cdot (2^{32})^8 \cdot 2^{32} \cdot 2^{32} \cdot 2^{32} \cdot 2 \cdot 2^{75.2} = 2^{358.9}$.

5. If a subkey guess is suggested by at least four quartets, then conclude that it is the right one.

Therefore, the total time complexity which is dominated by step 2 is approximately 2^{423} SHACAL-1 encryptions and the total data complexity of the attack is $2^{159.8}$ related-key chosen plaintexts which are encrypted under four related keys.

Table 5.2: The First Related-Key Differential for SHACAL-1

Round i	Δk_i	ΔA_i	ΔB_i	ΔC_i	ΔD_i	ΔE_i	Probability
0	e_1	0	0	e_{31}	e_{31}	e_{31}	2^{-1}
1	e_6	e_1	0	0	e_{31}	e_{31}	2^{-2}
2	$e_{1,31}$	0	e_1	0	0	e_{31}	2^{-2}
3	e_{31}	0	0	e_{31}	0	0	2^{-1}
4	$e_{1,31}$	0	0	0	e_{31}	0	2^{-2}
5	$e_{6,31}$	e_1	0	0	0	e_{31}	2^{-1}
6	0	0	e_1	0	0	0	2^{-2}
7	$e_{6,31}$	e_1	0	e_{31}	0	0	2^{-2}
8	e_{31}	0	e_1	0	e_{31}	0	2^{-3}
9	e_6	e_1	0	e_{31}	0	e_{31}	2^{-2}
10	e_{31}	0	e_1	0	e_{31}	0	2^{-3}
11	e_6	e_1	0	e_{31}	0	e_{31}	2^{-2}
12	$e_{1,31}$	0	e_1	0	e_{31}	0	2^{-3}
13	0	0	0	e_{31}	0	e_{31}	2^{-1}
14	e_{31}	0	0	0	e_{31}	0	2^{-1}
15	e_{31}	0	0	0	0	e_{31}	1
16	0	0	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
25	0	0	0	0	0	0	1
26	e_2	0	0	0	0	0	2^{-1}
27	e_7	e_2	0	0	0	0	2^{-1}
28	e_2	0	e_2	0	0	0	2^{-1}
29	$e_{0,3}$	0	0	e_0	0	0	2^{-2}
30	$e_{0,8}$	e_3	0	0	e_0	0	2^{-2}
31	$e_{0,3}$	0	e_3	0	0	e_0	2^{-2}
32	$e_{1,4}$	0	0	e_1	0	0	2^{-2}
33	$e_{1,9}$	e_4	0	0	e_1	0	2^{-2}
34	-	0	e_4	0	0	e_1	-

Table 5.3: The Second Related-Key Differential for SHACAL-1

Round i	Δk_i	ΔA_i	ΔB_i	ΔC_i	ΔD_i	ΔE_i	Probability
34	$e_{1,30}$	0	e_1	e_{31}	0	$e_{30,31}$	2^{-2}
35	e_1	0	0	e_{31}	e_{31}	0	2^{-1}
36	e_6	e_1	0	0	e_{31}	e_{31}	2^{-1}
37	$e_{1,31}$	0	e_1	0	0	e_{31}	2^{-1}
38	e_{31}	0	0	e_{31}	0	0	1
39	$e_{1,31}$	0	0	0	e_{31}	0	2^{-1}
40	$e_{6,31}$	e_1	0	0	0	e_{31}	2^{-2}
41	0	0	e_1	0	0	0	2^{-2}
42	$e_{6,31}$	e_1	0	e_{31}	0	0	2^{-2}
43	e_{31}	0	e_1	0	e_{31}	0	2^{-3}
44	e_6	e_1	0	e_{31}	0	e_{31}	2^{-2}
45	e_{31}	0	e_1	0	e_{31}	0	2^{-3}
46	e_6	e_1	0	e_{31}	0	e_{31}	2^{-2}
47	$e_{1,31}$	0	e_1	0	e_{31}	0	2^{-3}
48	0	0	0	e_{31}	0	e_{31}	2^{-1}
49	e_{31}	0	0	0	e_{31}	0	2^{-1}
50	e_{31}	0	0	0	0	e_{31}	1
51	0	0	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
60	0	0	0	0	0	0	1
61	e_2	0	0	0	0	0	2^{-1}
62	e_7	e_2	0	0	0	0	2^{-1}
63	e_2	0	e_2	0	0	0	2^{-1}
64	$e_{0,3}$	0	0	e_0	0	0	2^{-2}
65	$e_{0,8}$	e_3	0	0	e_0	0	2^{-2}
66	$e_{0,3}$	0	e_3	0	0	e_0	2^{-2}
67	$e_{1,4}$	0	0	e_1	0	0	2^{-2}
68	$e_{1,9}$	e_4	0	0	e_1	0	2^{-2}
69	-	0	e_4	0	0	e_1	-

5.4 Related-Key Impossible Boomerang Attack

Impossible boomerang attack which is a new extension of differential cryptanalysis was proposed by Lu [31] . It is a combination of impossible differential attack and boomerang attack. Then, this attack is combined with related-key scenario and it is called related-key impossible boomerang attack.

To describe the related-key impossible boomerang attack, consider the block cipher $E : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ as a cascade of two subciphers E_0 and E_1, i.e. $E = E_1 \circ E_0$ such that for E_0 there are two differentials $\alpha \rightarrow \beta$ and $\alpha' \rightarrow \beta'$ with probability 1 under keys K_1 and K_2 and for $(E_1)^{-1}$ there are two differentials $\delta \rightarrow \gamma$ and $\delta' \rightarrow \gamma'$ with probability 1 under keys K_3 and K_4 and intermediate differences satisfy the condition $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$. Related-key impossible boomerang distinguisher is based on the following theorem:

Theorem 5.4.1 *Let P_1, P_2, P_3 and P_4 be n-bit plaintext blocks where $P_2 = P_1 \oplus \alpha$ and $P_3 = P_4 \oplus \alpha'$. Assume that there are two differentials $\alpha \rightarrow \beta$ and $\alpha' \rightarrow \beta'$ with probability 1 for E_0 under keys K_1 and K_2 and there are two differentials $\delta \rightarrow \gamma$ and $\delta' \rightarrow \gamma'$ with probability 1 for $(E_1)^{-1}$ under keys K_3 and K_4 and intermediate differences satisfy the condition $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$. Then, the following two equations can not hold, simultaneously:*

$$E_{K_1}(P_1) \oplus E_{K_3}(P_3) = \delta \tag{5.1}$$

$$E_{K_2}(P_2) \oplus E_{K_4}(P_4) = \delta' \tag{5.2}$$

Proof. Assume that Equations 5.1 and 5.2 both hold, then by the differentials above,

$$\begin{aligned}
&\beta' = E_0^{K_3}(P_3) \oplus E_0^{K_4}(P_4) \\
&= (E_0^{K_3}(P_3) \oplus E_0^{K_1}(P_1)) \oplus (E_0^{K_1}(P_1) \oplus E_0^{K_2}(P_2)) \oplus (E_0^{K_2}(P_2) \oplus E_0^{K_4}(P_4)) \\
&= ((E_1^{K_3})^{-1}(E^{K_3}(P_3)) \oplus (E_1^{K_1})^{-1}(E^{K_1}(P_1)) \oplus (E_0^{K_1}(P_1) \oplus E_0^{K_2}(P_2)) \oplus ((E_1^{K_2})^{-1}(E^{K_2}(P_2)) \oplus \\
&(E_1^{K_4})^{-1}(E^{K_4}(P_4)) \\
&= \gamma \oplus \beta \oplus \gamma'.
\end{aligned}$$

In the equation above, we get that $\beta = \gamma \oplus \beta \oplus \gamma'$, however, this contradicts with the condition

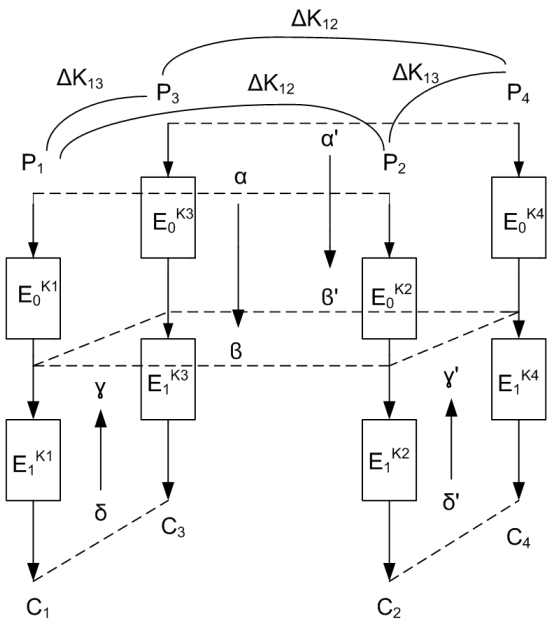

Figure 5.7: Related-key impossible boomerang distinguisher satisfying the condition $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$

that $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$. Then, proof of the theorem is done. ■

Consequently, it is concluded that the input difference pair (α, α') can never cause the output difference pair (δ, δ') with probability 1, such a differential is called *a related-key impossible boomerang distinguisher* depicted in figure 5.7 which can be written as $(\alpha, \alpha') \nrightarrow (\delta, \delta')$.

Using related-key impossible boomerang distinguisher, the key recovery attack can usually be constructed as follows:

1. Consider the block cipher $E : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ as a cascade of four subciphers $E = E^f \circ E^1 \circ E^0 \circ E^i$, where $E^1 \circ E^0$ is the impossible boomerang distinguisher, E^i includes the rounds before $E^1 \circ E^0$ and E^f includes the rounds after $E^1 \circ E^0$.

2. Guess necessary key bits/bytes in E^i and E^f, and check whether the quartet of known plaintext/ciphertext pairs $(((P_1, C_1), (P_2, C_2))((P_3, C_3), (P_4, C_4)))$ satisfy the following four equations:

$$E_i^{K_1}(P_1) \oplus E_i^{K_2}(P_2) = \alpha$$
$$E_i^{K_3}(P_4) \oplus E_i^{K_4}(P_4) = \alpha'$$

$(E_f^{K_1})^{-1}(C_1) \oplus (E_f^{K_3})^{-1}(C_3) = \delta$
$(E_f^{K_2})^{-1}(C_2) \oplus (E_f^{K_4})^{-1}(C_4) = \delta'$.

3. If a guartet satisfies the above four conditions, then conclude that guessed subkey quartet (K_1, K_2, K_3, K_4) is wrong and discard it. By this way, taking sufficient number of quartets of known plaintext/ciphertext pairs make the wrong keys eliminate and leave the actual key.

5.4.1 A 6-Round Related-Key Impossible Boomerang Distinguisher for AES-192

Related-key impossible boomerang attack on 8-round AES-192 is the first attempt to use this method to a block cipher by Lu [31]. It is applied to 8-round AES-192 with a running time of 2^{160} 8-round AES encryptions. This attack is based on 6-round related-key impossible boomerang distinguisher which is explained in detail in the following subsection. In this section we only mention the related-key distinguisher which is a new type of distinguisher for AES-192.

If the key differences between related keys are chosen as $K_1 \oplus K_2 = K_3 \oplus K_4 = (\Delta K_{col(0)}, \Delta K_{col(1)}, \ldots \Delta K_{col(5)}) = ((a,0,0,0),(0,0,0,0),(a,0,0,0),(0,0,0,0),(0,0,0,0),(0,0,0,0))$, where a 8-bit fixed non-zero value, then subkey differences used in the attack are given in Table 5.4:

Table 5.4: Subkey Differences

Round(i)	$\Delta k_{i,Col(0)}$	$\Delta k_{i,Col(1)}$	$\Delta k_{i,Col(2)}$	$\Delta k_{i,Col(3)}$
0	(a,0,0,0)	(0,0,0,0)	(a,0,0,0)	(0,0,0,0)
1	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
2	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
3	(a,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
4	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
5	(a,0,0,0)	(a,0,0,0)	(a,0,0,0)	(a,0,0,0)
6	(a,0,0,b)	(0,0,0,b)	(a,0,0,b)	(0,0,0,b)
7	(a,0,0,b)	(0,0,0,b)	(a,0,c,b)	(a,0,c,0)
8	(0,0,c,b)	(0,0,c,0)	(a,0,c,b)	(a,0,c,0)

The distinguisher is built combining four differentials shown in Figure 5.8:

The first differential $\alpha \rightarrow \beta$ **for** E_0 is :

$((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0)) \rightarrow ((f_1,e_2,e_3,e_4),(f_5,e_6,e_7,e_8),(f_9,e_{10},e_{11},e_{12}),$

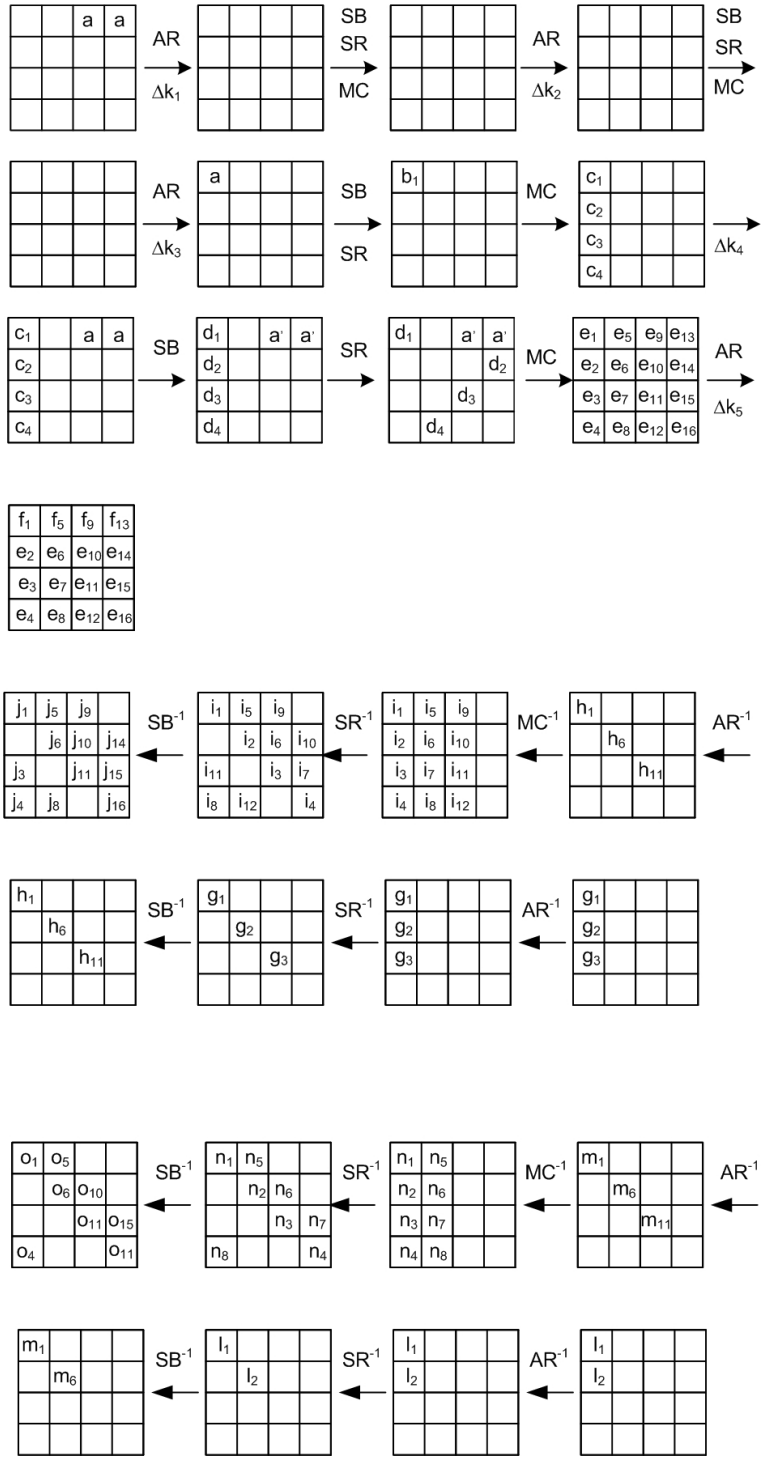

Figure 5.8: Differentials with probability 1 for AES-192

$(f_{13}, e_{14}, e_{15}, e_{16}))$,

The second differential $\alpha' \rightarrow \beta'$ **for** E_0 is:

$((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0)) \rightarrow ((f'_1, e'_2, e'_3, e'_4),(f'_5, e'_6, e'_7, e'_8),(f'_9, e'_{10}, e'_{11}, e'_{12}),$
$(f'_{13}, e'_{14}, e'_{15}, e'_{16}))$,

The first differential $\delta \rightarrow \gamma$ **for** E_1 is :

$((g_1, g_2, g_3, 0),(0,0,0,0),(0,0,0,0),(0,0,0,0)) \rightarrow ((j_1, 0, j_3, j_4),(j_5, j_6, 0, j_8),(j_9, j_{10}, j_{11}, 0),$
$(0, j_{14}, j_{15}, j_{16})$,

The second differential $\delta' \rightarrow \gamma'$ **for** E_1 is:

$((l_1, l_2, 0, 0),(0,0,0,0),(0,0,0,0),(0,0,0,0)) \rightarrow ((o_1, 0, 0, o_4),(o_5, o_6, 0, 0),(0, o_{10}, o_{11}, 0),$
$(0, 0, o_{15}, o_{16})$,

Then, it is observed that these differentials can form the following 6-round impossible boomerang distinguisher for AES-192:

$(((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0)),((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0))) \nrightarrow$
$(((g_1, g_2, g_3, 0),(0,0,0,0),(0,0,0,0),(0,0,0,0)),((l_1, l_2, 0, 0),(0,0,0,0),(0,0,0,0),(0,0,0,0))$.

This distinguisher works, because:

By the property of the MC operation, in the first differential $\alpha \rightarrow \beta$ for E_0, we have the equalities $e_2 = d_1$ and $e_3 = d_1$ and in the second differential $\alpha' \rightarrow \beta'$ for E_0, we have the equalities $e'_2 = d'_1$ and $e'_3 = d'_1$. Since the branch number of MC operation is 5 [**?**], then $i_{11} \neq 0$. Therefore, $j_3 \neq 0$. This is a direct consequence of one to one property of SB operation. Notice that the second bytes of γ and γ' are both 0 and the third bytes of γ and γ' are j_3 and 0, respectively. Then, XOR of the second bytes of the four differentials $\beta \oplus \beta' \oplus \gamma \oplus \gamma'$ is $e_2 \oplus e'_2 = d_1 \oplus d'_1$ and XOR of the third bytes of the four differentials $\beta \oplus \beta' \oplus \gamma \oplus \gamma'$ is $e_3 \oplus e'_3 \oplus j_3 = d_1 \oplus d'_1 \oplus j_3$. Since it is impossible to make two values $d_1 \oplus d'_1$ and $d_1 \oplus d'_1 \oplus j_3$ zero at the same time, the distinguisher works ($\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$).

This distinguisher given above is used to attack on 8-round AES-192 and for the details of the attack, refer to [31].

CHAPTER 6

RELATED-KEY DIFFERENTIAL-LINEAR CRYPTANALYSIS

Differential-linear cryptanalysis which is a combination of differential and linear cryptanalysis was proposed by Langford and Hellman in 1994 [49]. They show that concatenating a differential characteristic and a linear approximation is possible and allows to attack many block ciphers. This cryptanalytic method is firstly applied to 8-round DES in the same paper in which the attack is presented [49]. Then, in 2002, Biham et al. proposed enhanced version of the attack and applied to DES and COCONUT98 [50]. Afterwards, differential-linear cryptanalysis was combined with related-key scenario and firstly used to attack on AES-192 [51].

This chapter is mainly about the related-key differential-linear cryptanalysis. The structure of the chapter is as follows: Section 6.1 gives detailed description of related-key differential-linear attack. Then, Section 6.2 exemplifies the related-key differential-linear on AES-192.

6.1 Overview of the Attack

Differential-linear attack is a combination of differential and linear attacks in a way that it exploits both a differential characteristics and a linear approximation too. To mount a differential-linear attack, the attacker first constructs a differential characteristics, then constructs a linear approximation which is attached to the differential characteristic. By this way, the attacker derives a longer distinguisher which allows to attack on more rounds of block ciphers which are resistant to both differential and linear attacks.

To mount related-key differential-linear attack, firstly we need a related-key differential-linear

distinguisher. The construction of a related-key differential-linear distinguisher is based on the following proposition:

Proposition 6.1.1 *Consider the block cipher* $E : \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$ *as a cascade of two subciphers* E_0 *and* E_1 *such that* $E = E_0 \circ E_1$. *Assume that there exist a differential* $\alpha \rightarrow \beta$ *with probability* p *for* E_0 *and a linear approximation* $\gamma \rightarrow \delta$ *with bias* ϵ *for* E_1. *Let a plaintext pair* (P, P^*) *satisfies* $P^* = P \oplus \alpha$, *then*

$$Pr[(\delta \cdot E_{K_1}(P) \oplus \delta \cdot E_{K_2}(P^*)) = \gamma \cdot \beta] = \frac{1}{2} + 2p\epsilon^2.$$

Proof. Assume that a plaintext pair (P, P^*) satisfies the difference α, i.e $P^* = P \oplus \alpha$, then by the differential $\alpha \rightarrow \beta$ for E_0, the equation $E^0_{K_1}(P) \oplus E^0_{K_2}(P^*) = \beta$ holds with probability p and by the linear approximation $\gamma \rightarrow \delta$ for E_1, the equation $\gamma \cdot E^0_{K_1}(P) = \delta \cdot E_{K_1}(P)$ is satisfied with bias ϵ and the equation $\gamma \cdot E^0_{K_2}(P^*) = \delta \cdot E_{K_2}(P^*)$ is satisfied with bias ϵ. If the equations $\gamma \cdot E^0_{K_1}(P) = \delta \cdot E_{K_1}(P)$ and $\gamma \cdot E^0_{K_2}(P^*) = \delta \cdot E_{K_2}(P^*)$ are combined, then the equation $\gamma \cdot \underbrace{(E^0_{K_2}(P^*) \oplus E^0_{K_1}(P))}_{\beta} = \gamma \cdot \beta = (\delta \cdot E_{K_1}(P)) \oplus (\delta \cdot E_{K_2}(P^*))$ is obtained with a probability of:

$$p[(\frac{1}{2} + \epsilon)(\frac{1}{2} + \epsilon)(\frac{1}{2} - \epsilon)(\frac{1}{2} - \epsilon)] = p(\frac{1}{2} + 2\epsilon^2)$$

It is assumed that $E^0_{K_1}(P) \oplus E^0_{K_2}(P^*) = \beta$ holds with probability p, however even if $E^0_{K_1}(P) \oplus E^0_{K_2}(P^*) \neq \beta$ holds with probability $(1 - p)$ and it is assumed that the value $(\delta \cdot E_{K_1}(P)) \oplus (\delta \cdot E_{K_2}(P^*))$ is a random distribution, the probability of $\gamma \cdot \beta = (\delta \cdot E_{K_1}(P)) \oplus (\delta \cdot E_{K_2}(P^*))$ is

$$(1 - p) \cdot \frac{1}{2} = \frac{1}{2} - \frac{p}{2}.$$

Therefore, if the above two cases are both considered, then the probability of Equation $\gamma \cdot \beta = (\delta \cdot E_{K_1}(P)) \oplus (\delta \cdot E_{K_2}(P^*))$

$$p(\frac{1}{2} + 2\epsilon^2) + (1 - p) \cdot \frac{1}{2} = \frac{1}{2} + \underbrace{2p\epsilon^2}_{\epsilon'}.$$

If E is a random permutation, it is expected that the equation $\gamma \cdot \beta = \delta \cdot (E_{K_1}(P)) \oplus \delta \cdot (E_{K_2}(P^*))$ holds with probality $\frac{1}{2}$. Therefore, if bias $\epsilon' = 2p\epsilon^2$ of Equation $\gamma \cdot \beta = (\delta \cdot E_{K_1}(P)) \oplus (\delta \cdot E_{K_2}(P^*))$ is high enough, then the cipher E can be distinguished from a random cipher permutation by

using the related-key differential-linear distinguisher.

In the following section, an application of this attack to reduced version of AES by Zhang et al. [51] is described .

6.2 Related-Key Differential-Linear Cryptanalysis of Reduced-Round AES-192

In this section, a related-key differential linear attack on 7-round AES-192 presented by Zhang et al. [51] will be covered. The attack utilizes 5-round related-key differential-linear distinguisher which is given in detail in the following subsection.

6.2.1 Notations

Notations used in the description of the distinguisher and the attack is as follows:

k_i: The subkey of round i, $0 \leq i \leq 3$
$k_{i,Col(j)}$: The j^{th} column of k_i, $0 \leq i \leq 3, 0 \leq j \leq 3$
x_i^I: The input to round i
x_i^S: The output of SubByte operation in round i
x_i^R: The output of ShiftRow operation in round i
x_i^M: The output of MixColumn operation in round i
x_i^O: The output of round i
$x_{i,col(j)}$: The j^{th} column of x_i , $0 \leq i \leq 3, 0 \leq j \leq 3$
$(x_i)_j$: The j^{th} byte of x_i, $0 \leq i \leq 3, 0 \leq j \leq 3$
?: Any byte difference N: Non-zero difference

Note that for the description of AES-192 and related notations, refer to Chapter 4.

6.2.2 A 5-Round Related-Key Differential-Linear Distinguisher for AES-192

A 5-round differential-linear distinguisher is built by concatenating a 4-round related-key differential characteristics with 1-round related-key linear approximation. The difference

between two related keys K and K^* is chosen as ((0,0,0,0), (0,0,0,0), (a,0,0,0), (a,0,0,0), (0,0,0,0), (0,0,0,0)). Then, the subkey differences are used in the attack are computed and given in Table 6.1.

Table 6.1: Subkey Differences

Round(i)	$\Delta k_{i,Col(0)}$	$\Delta k_{i,Col(1)}$	$\Delta k_{i,Col(2)}$	$\Delta k_{i,Col(3)}$
0	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
1	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
2	(a,0,0,0)	(0,0,0,0)	(0,0,0,0)	(0,0,0,0)
3	(0,0,0,0)	(0,0,0,0)	(a,0,0,0)	(a,0,0,0)
4	(a,0,0,0)	(a,0,0,0)	(0,0,0,b)	(0,0,0,b)
5	(a,0,0,b)	(0,0,0,b)	(a,0,0,b)	(0,0,0,b)
6	(0,0,c,b)	(0,0,c,0)	(a,0,c,b)	(a,0,c,0)
7	(0,0,c,b)	(0,0,c,0)	(0,d,c,b)	(0,d,0,b)
8	(a,d,c,0)	(0,d,0,0)	(0,d,c,b)	(0,d,0,b)

A 4-round differential depicted in Figure 6.1 is built as follows:

1. Choose the difference between the plaintext pairs (P, P^*) as $P \oplus P^* = ((0,0,0,0), (0,0,0,0), (a,0,0,0), (a,0,0,0))$, then the plaintext difference is cancelled by the whitening subkey difference Δk_0, so the difference in all bytes of input to round 1 Δx_1^I becomes zero.

2. This zero difference is preserved through SB, SR and MC operations in the first and the second rounds until adding the subkey difference Δk_2. Then, the first byte of Δx_3^I will become a nonzero value.

3. After MC operation in round 3, this difference diffuses to all column. Let's call the non-zero byte position zero of the fourth round as ϵ. Then, after MC operation, the first column of Δx_4^M will be $(02 \cdot \epsilon, \epsilon, \epsilon, 03 \cdot \epsilon)$.

4. Finally, after applying the key addition, equal differences in byte 1 and 2 of Δx_4^O are obtained. Therefore, $\Delta(x_4^O)_1 = \Delta(x_4^O)_2 \neq 0$ holds with probability 1.

Then, 1-round linear approximation is attached to the end of 4-round related-key differential distinguisher. In SB operation in round 5, a linear approximation is used. Since there are $(2^8 - 1)$ possible nonzero value for difference ϵ, there are $2^{16} \cdot (2^8 - 1)$ possible values for

quartets $((x_4^O)_1, (x_4^O)_1^*, (x_4^O)_2, (x_4^O)_2^*)$ satisfying the equation $(x_4^O)_1 \oplus (x_4^{*O})_1 = (x_4^O)_2 \oplus (x_4^{*O})_2 \neq 0$. Authors computed that for every possible 8-bit linear mask $\lambda \in \left\{1, 2, \ldots, 2^8 - 1\right\}$, the equation

$$\lambda \cdot \left\{(x_5^S)_1 \oplus (x_5^{*S})_1 \oplus (x_5^S)_2 \oplus (x_5^{*S})_2\right\} = 0$$

is satisfied with a bias of 2^{-9}.

Then, after SR operation in round 5, the following equation is satisfied again with a bias of 2^{-9}:

$$\lambda \cdot \left\{(x_5^R)_{13} \oplus (x_5^{*R})_{13} \oplus (x_5^R)_{10} \oplus (x_5^{*R})_{10}\right\} = 0$$

Finally, after AR operation in round 5, a 5-round related-key differential-linear distinguisher is built with a bias of 2^{-9}:

$$\lambda \cdot \left\{(x_5^W)_{13} \oplus (x_5^{*W})_{13} \oplus (x_5^W)_{10} \oplus (x_5^{*W})_{10}\right\} = 0$$

or equivalently

$$\lambda \cdot \left\{\Delta(x_5^W)_{13} \oplus \Delta(x_5^{*W})_{10}\right\} = 0$$

.

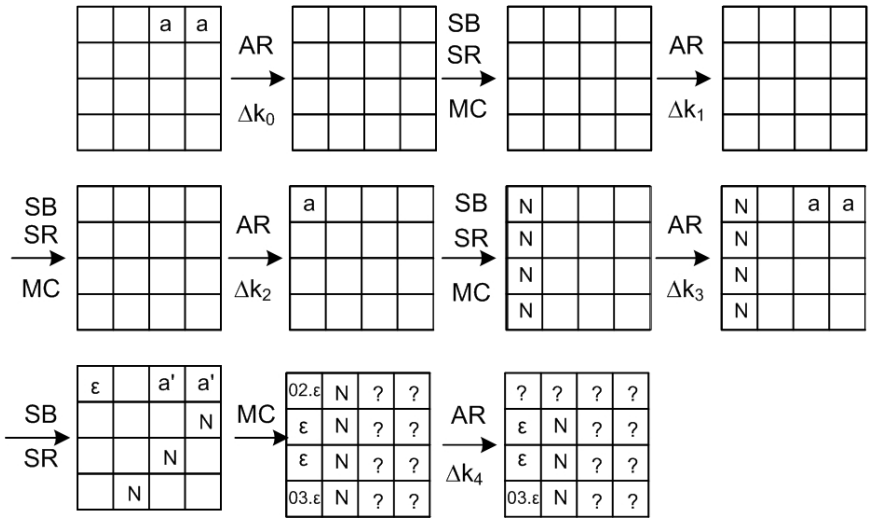

Figure 6.1: A 4-round differential for AES

6.2.3 A 7-Round Related Key Differential-Linear Attack on AES-192

Main idea of this attack is to apply the 5-round related-key differential-linear distinguisher given above starting from the first round to guess some key bytes of the sixth and the seventh rounds for partial decryption, then to use the linear approximation to find the key.

To mount the attack, assume that the values of a,b,c and d are known. Then, the attack algorithm goes as follows:

- Form two sets of plaintexts $\mathcal{P}_1$ and $\mathcal{P}_2$ containing m plaintexts each such that each pair $P_1 \in \mathcal{P}_1$ and $P_2 \in \mathcal{P}_2$ satisfies the desired difference $P_1 \oplus P_2 = ((0,0,0,0),(0,0,0,0),(a,0,0,0),(a,0,0,0))$.

- Obtain the sets of C_1 and C_2 which are the corresponding ciphertexts of the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ under two related keys K_1 and K_2, respectively.

- Initialize an array of 2^{160} counters to zero.

- Guess all bytes of k_7 and bytes in positions 8,9,12,15 of w_6, then partially decrypt each ciphertext pair (C_1, C_2) and obtain the bytes $(x_5^W)_{10}$ and $(x_5^W)_{13}$.
  Check whether the equation $\lambda \cdot \left\{\Delta(x_5^W)_{10} \oplus \Delta(x_5^W)_{13}\right\} = 0$ is satisfied, where λ is fixed to 0x01. If this equation holds, increment the counter of the array by 1.

- If $|T_{max} - m/2| > |T_{min} - m/2|$, then conclude that the key candidate corresponding to T_{max} where T_{max} is the highest entry is the actual key. Otherwise, conclude that the key candidate corresponding to T_{min} where T_{min} is the lowest entry is the actual key.

6.2.4 Attack Complexity

It is assumed that the values a,b,c and d are known. If the value of a is fixed a certain value, then applying SB operation, there are $2^7 - 1 = 127$ different possible values for b. Due to the structure of the key schedule, the value of c is derived from the value of b and the value of d is derived from c. For this reason, if the attacker fixes the value of a to a certain value, the attack needs to be repeated only for 127 possible values of b.

Data Complexity: Since the bias of the distinguisher is 2^{-9}, therefore the attack needs $8 \cdot (\frac{1}{2^{-9}})^2 = 2^{21}$ plaintext-ciphertext pairs due to the Matsui's rule of thumb.

Time Complexity: Since 20 bytes of subkeys are guessed, 2-round decryption is done during the attack and the attack is repeated for each 127 possible values of b, this attack has the time complexity of $2^{160} \cdot 2^{22} \cdot \frac{2}{7} \approx 2^{180}$ 7-round AES encryptions.

CHAPTER 7

SLIDE ATTACKS

Nowadays, block ciphers are designed to resist to the most powerful attacks such as differential and linear attacks. In addition to this, most of newly designed block ciphers have high speed and simplicity in hardware and software implementations to be used in sensitive applications such as RFID systems and sensor networks, etc. For this reason, such block ciphers have simple encryption and key scheduling algorithms. However, the simplicity can cause the cipher vulnerable to cryptanalytic attacks. Therefore, the designers increase the number of rounds of such block ciphers to make them secure. However, having high number of rounds cannot always provide security against all attacks. The block cipher can still be insecure and can be broken by using slide attack which is independent of number of rounds.

This chapter is mainly about Slide Attacks and its extensions: Advanced Slide Attacks and Improved Slide Attacks.

7.1 Slide Attack

Slide attack was proposed by Biryukov and Wagner in 1999 [52, 53]. This attack is inspired by Knudsen's work [54] and the attack on LOKI89 and LOKI91 proposed by Biham [8], thus, it is closely related to the related-key attack. Like related-key attack, slide attack exploits both weaknesses of the key schedule and the encryption algorithm of the given block cipher. However, slide attack is different from related-key attack in a sense that while related-key attack uses two or more keys to make encryptions, slide attack uses only one key. Besides this, slide attack is applicable to iterative block ciphers which have self-similarity. More explicitly,

if a block cipher has an iterative structure and a periodic key schedule with periodicity p, then the cipher repeats itself in every p-rounds. Such a cipher is called p-round self-similar and vulnerable to slide attack due to its self-similarity.

7.1.1 A Typical Slide Attack

Assume that the block cipher E can be written as a composition of r identical permutations such that $E = F \circ F \circ \cdots \circ F = F^r$ where F is a permutation which excepts a round key K and the output of the previous permutation or plaintext as an input. The permutation F can consist of one ore more rounds of the cipher. The crucial idea of a typical slide attack is to slide a copy of (a permutation) encryption process against another copy of encryption process so that each process is one round of phase.

Proposition 7.1.1 *Let the pair (P_0, C_0) and (P_0^*, C_0^*) be a known plaintext-ciphertext pair for the given block cipher and suppose that the relation $P_1 = P_0^*$ holds, then the relation $P_r = P_{r-1}^*$ is obtained with probability 1 where P_i and P_i^* are the outputs of round i.*

Proof. This can be proved by the Induction Hypothesis: Assume that the relation $P_k = P_{k-1}$ holds for some k, then $P_{k+1} = F(P_k, K) = F(P_{k-1}^*, K) = P_k^*$ which proves the relation $P_r = P_{r-1}^*$. ■

In this attack, the attacker aims to find a pair (P, C) and (P^*, C^*) satisfying $F(P, K) = P^*$, such a pair is called a *slid pair*. Then, the relation $F(C, K) = C^*$ is obtained for free. Slide attack is pictorially depicted in Figure 7.1.

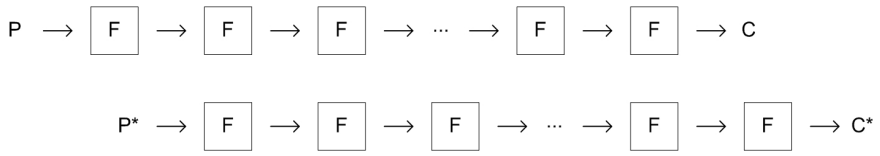

Figure 7.1: Typical slide attack

To perform the attack, the attacker needs to find a slid pair to derive the secret key K. However, finding a slid pair is not trivial, since $F(P, K)$ can not be determined. But, slid equations

$F(P, K) = P^*$ and $F(C, K) = C^*$ are not both satisfied if (P, C) and (P^*, C^*) do not constitute a slid pair which gives an approach to identify a slid pair. Afterwards, for each pair (P, C) and (P^*, C^*) solve Equations $F(P, K) = P^*$ and $F(C, K) = C^*$ until to find K which verifies the equations. Once s solution K has been found for the equations, the pair is probably is a slid pair and the solution K is the actual value of K. However, the permutation F should be a weak to allow us to extract key K. Such a permutation is called a *weak* permutation and the cipher should have a weak permutation F.

The cipher is broken by taking $2^{n/2}$ known plaintext-ciphertexts (P^i, C^i) where n is a block size. By the birthday paradox, it is expected to find a slid pair (P^i, C^i) and (P^j, C^j) for some i, j which satisfies the relation $F(P, K) = P^*$. A typical slide attack needs $O(2^{n/2})$ known plaintext-ciphertexts and since there are $O(2^n)$ possible plaintext-ciphertext pairs, the time complexity of the attack is $O(2^n)$.

7.1.2 Slide Attack on Feistel Ciphers

Slide attack is applied to Feistel ciphers with reduced time and data complexities thanks to their structures in two ways: Known plaintext-ciphertext attack and chosen plaintext-ciphertext attack.

To explain slide attack on Feistel ciphers, we consider a generic r-round Feistel Network with one round self-similarity (All round keys are the same). Notice that this attack can be applicable to any feistel ciphers with infinite number of rounds. Sliding is shown on an r-round Feistel cipher as depicted in Figure 7.2.

In the case of Feistel ciphers, a pair (P, C) and (P^*, C^*) form a slid pair if the relation between plaintexts $F(P_L, P_R) = (P_R, P_L \oplus f(P_R \oplus K)) = (P_L^*, P_R^*)$ holds which makes the relation between the ciphertexts $F(C_R, C_L) = (C_R, C_L \oplus f(C_R \oplus K)) = (C_L^*, C_R^*)$ hold for free. Then, the slid equations for Feistel ciphers are:

$$P_L^* = P_R \tag{7.1}$$

$$P_R^* = P_L \oplus F(P_R, K) \tag{7.2}$$

$$C_R^* = C_L \tag{7.3}$$

$$C_L^* = C_R \oplus F(C_R^*, K) \tag{7.4}$$

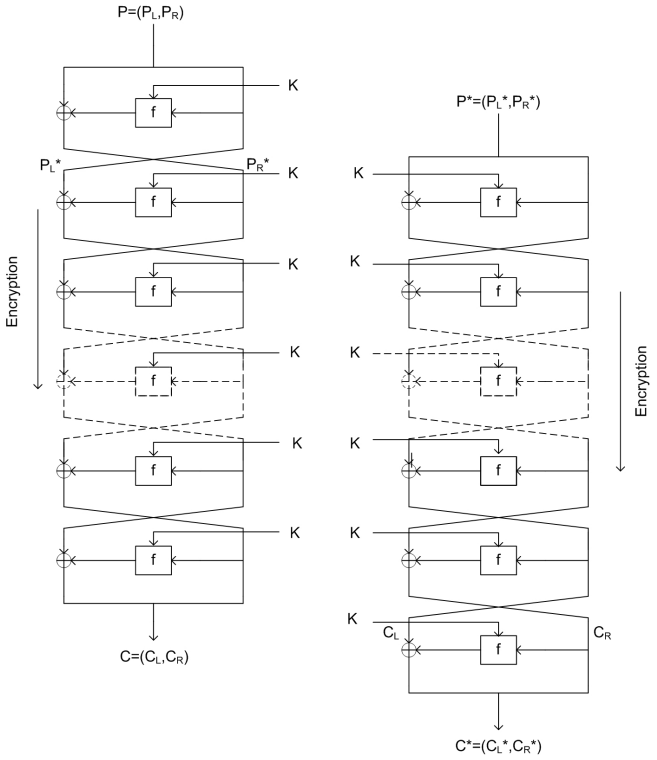

Figure 7.2: Typical slide attack on a generic r-round Feistel cipher with one round self-similarity

Known Plaintext-Ciphertext Attack: To mount attack, take a set of $2^{n/2}$ known plaintext-ciphertexts (P^i, C^i) which constitute 2^n plaintext-ciphertext pairs. By the Birthday Paradox, we expect to get only one slid pair. Then, due to the Feistel construction, Equality 7.2 which makes $\frac{n}{2}$-bit filtering condition on a slid pair should be satisfied. On the other hand, there is also $\frac{n}{2}$-bit filtering condition on a slid pair for ciphertext pairs, that is $C_R = C_L^*$. In total, there are n-bit filtering condition on a slid pair. Thus, If we sort all $2^{n/2}$ plaintext-ciphertexts into a hash table and check the equalities $P_R = P_L^*$ and $C_R = C_L^*$, then we expect that there are left a slid pair and a false alarm. A slid pair that we have found allows us to derive correct key value K by solving Equations 7.2 and 7.4. In addition, it is easy to eliminate the wrong key

suggested by a false alarm by just trying two or three plaintext-ciphertexts pair once key K have found. This attack is implemented with $O(2^{n/2})$ data complexity, $O(2^{n/2})$ offline work.

Chosen Plaintext-Ciphertext Attack: In the case of Feistel ciphers, if the attacker can choose carefully plaintexts satisfying $P_R = P_L^*$, then the required data is reduced from $2^{n/2}$ to $2^{n/4}$ plaintext-ciphertexts which constitute about $2^{n/2}$ pairs of plaintexts. Then, due to the slide attack's assumption, the attacker picks a set of $2^{n/4}$ plaintexts of the form (P_L^i, A) and another set of $2^{n/4}$ plaintexts of the form (A, P_R^j) where A is $\frac{n}{2}$-bit fixed value to satisfy Equation 7.1. Since the required Equation 7.4 has $\frac{n}{2}$-bit filtering condition, it eliminates $2^{n/2}$ of the pairs. Hence, it is expected to find one slid pair which allows to extract the correct key value of K from Equations 7.2 and 7.4.

7.2 Advanced Slide Techniques

Biryukov and Wagner upgraded the slide attack and presented two new techniques: complementation slide and sliding with a twist [55].

These techniques optimize a typical slide attack in way that in a typical slide attack, if the cipher have two rounds (or more) self-similarity, one should slide by two rounds (or more), on the contrary, in advanced slide attacks, sliding by one round is enough to mount attack. This reduces the complexity of a typical slide attack. Then, in 2005, Phan [56] introduced a more effective technique called Realigning Slide Attack which enables to attack block ciphers having irregular key schedules.

7.2.1 Complementation Slide

In a typical slide attack, if the cipher is 2-round self similar, one must slide encryption process by two rounds which causes inefficient attacks. In an advanced technique, namely complementation slide attack, one can slide only by one round as in the case of a typical slide attack.

In this method, a pair (P, C) and (P^*, C^*) forms a slid pair if they satisfy the following equa-

tions:

$$(P_L^*, P_R^*) = (P_R \oplus \delta, P_L \oplus f(K_0 \oplus P_R) \oplus \delta) \quad (7.5)$$

$$(C_L^*, C_R^*) = (C_R \oplus \delta, C_L \oplus f(K_1 \oplus C_R \oplus \delta) \oplus \delta) \quad (7.6)$$

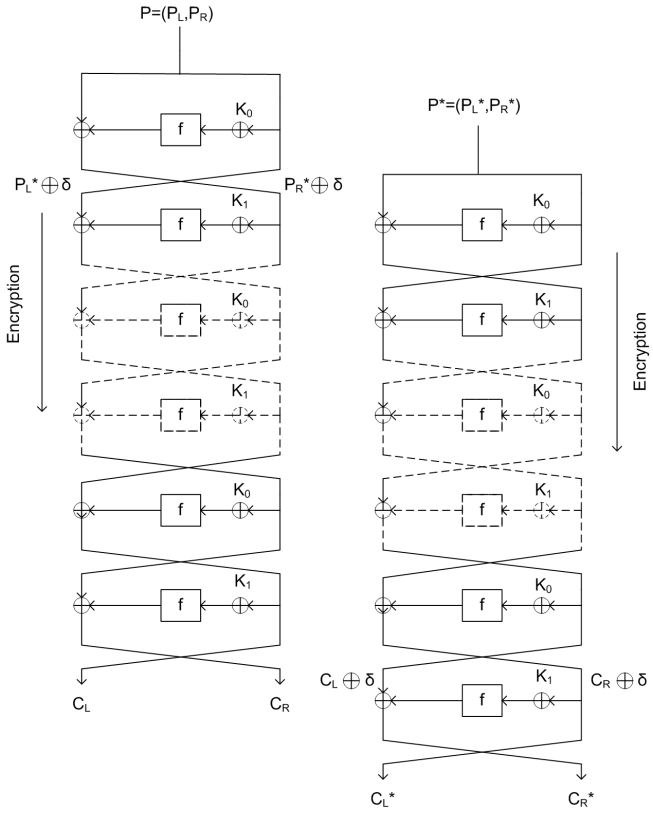

Figure 7.3: Complementation slide attack on a generic r-round Feistel cipher with two-round self-similarity

In this technique, instead of choosing a slid difference between plaintexts as zero, one can choose a slid difference as (δ, δ), where $\delta = K_0 \oplus K_1$. This causes the slid difference δ to be cancelled by the subkey difference δ, appear in every round and show up at the ciphertext differences.

By using slid Equations 7.5 and 7.6, this technique was applied to DESX which is an extension of DES. For the details of the attack, refer to [55].

7.2.2 Sliding with a Twist

This method is described by using a generic Feistel cipher which has two-round self-similarity. Unlike typical slide attack, one copy of encryption process is slided by one round against one copy of decryption process so that each process is one round out of phase which is called as the *twist*. This is reasonable, because in a generic Feistel cipher, encryption under the key $K = (K_0, K_1)$ is the same as decryption under $K = (K_1, K_0)$. Encryption from the second round follows the same way with the decryption starting one round before the last round. This come up with a slid pair satisfying the following relations:

$$(C_L^*, C_R^*) = (P_R, P_L \oplus f(K_0 \oplus P_R)), \tag{7.7}$$

$$(P_L^*, P_R^*) = (C_R, C_L \oplus f(K_0 \oplus C_R)). \tag{7.8}$$

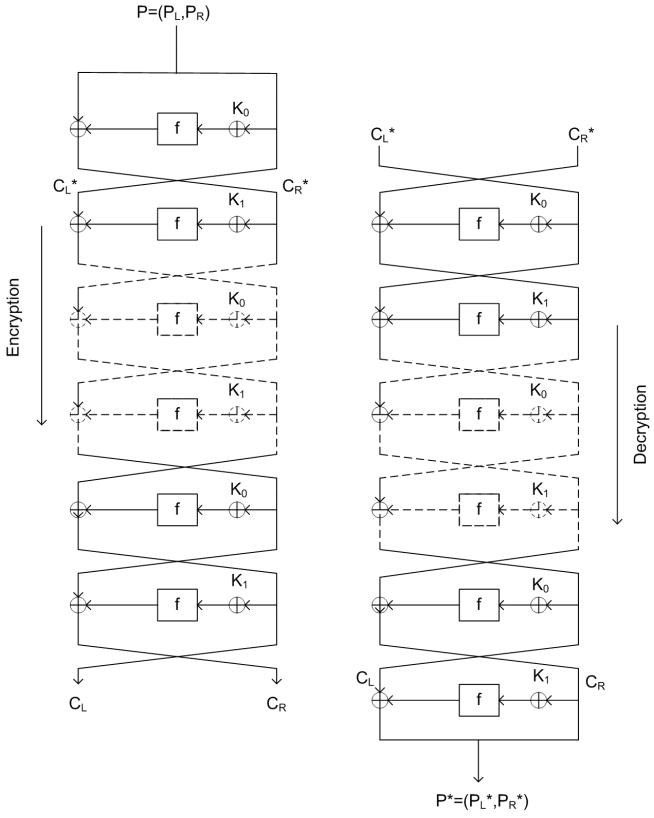

Figure 7.4: Sliding with a twist attack on a generic Feistel cipher with two-round self-similarity

From Equations 7.7 and 7.8, the relations $P_L^* = C_R$ and $C_R^* = P_R$ provide $\frac{n}{2} + \frac{n}{2} = n$-bit filtering condition on a slid pair. If a set of $2^{n/2}$ plaintexts-ciphertexts is taken which provides 2^n plaintext-ciphertext pairs, then it is expected to get only one slid pair. Recognizing a slid

pair is straightforward by sorting all $2^{n/2}$ data into a hash table which gives the real subkey value K_0.

Note that once one have found K_0, K_1 can also be found by using a typical slide attack. This attack can also be done by choosing plaintexts so that $P_R^* = P_R$. By this way, the data complexity of the attack is reduced from $2^{n/2}$ to $2^{n/4+1}$.

7.2.2.1 Cryptanalysis of DESX with Sliding Twist Technique

DESX which was proposed by Rivest in 1984 is a variant of DES. The design rational of DESX is to make DES more resistant to Brute Force attack. DESX has the same components as DES, however, different from DES, DESX has initial and final key whitenings. This property of DESX makes it vulnerable to sliding twist technique.

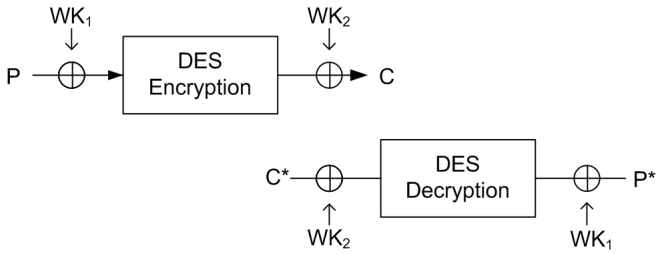

Figure 7.5: Twisting slide attack on full-round DESX

It is assumed that a slid pair (P, C) and (P^*, C^*) satisfies the relation $C \oplus C^* = WK_1$, then the equality

$$P^* = E^{-1}(C^* \oplus WK_2) \oplus WK_1 = E^{-1}(C) \oplus WK_1 \tag{7.9}$$

is obtained.

If Equation 7.9 is combined with the relation $P = E^{-1}(C^*) \oplus WK_1$, then we have the equality $WK_1 = E^{-1}(C^*) \oplus P = E^{-1}(C) \oplus P^*$ which gives a condition on a slid pair.
Then, the attack algorithm goes as follows:

- Take a set of $2^{32.5}$ known plaintext-ciphertexts (P^i, C^i) which constitutes 2^{64} plaintext-ciphertext pairs.

- Guess 56-bit secret key K of DES.
- For each guess of K, construct a lookup table and search a slid pair (P, C) and (P^*, C^*) that satisfies $E_K^{-1}(C^i) \oplus P^i = E_K^{-1}(C^j) \oplus P^j$ for some i, j.
- For each guess of K, it is expected to get a false slid pair which can be discarded easily and a true slid pair if the guessed key is correct. Once a slid pair have been found, whitening keys WK_1 and WK_2 can be identified by using the relations $C \oplus C^* = WK_1$ and $WK_1 = E^{-1}(C^*) \oplus P$ or $WK_1 = E^{-1}(C) \oplus P^*$. The accuracy of the candidate key of DESX (WK_1, K, WK_2) can be tested easily by using 2 or 3 pairs of plaintext-ciphertext pairs.

  This attack has $2^{32.5}$ known plaintexts as data complexity and $2^{56} \cdot 2^{32.5} = 2^{86.5}$ offline full round DES encryptions.

7.2.3 Realigning Slide

A typical slide attack and advanced slide attacks- complementation slide and twisting slide- can be applied to block ciphers which have self-similarity up to four rounds [52, 53, 55]. For this reason, to extent applicability of slide attack, Phan [56] introduced an advanced slide technique called **Realigning Slide** which enables to attack block ciphers having irregular key schedule. More explicitly, even if the block cipher have dissimilar rounds between the sliding rounds or self-similarity more than 4-round, sliding is still possible by realigning slide technique. To describe this technique, Phan presented a new attack on full round DES that will be mentioned in the next section.

7.2.3.1 Realigning Slide Attack on Full Round DES

Realigning Slide technique can be applied to full round DES with tweaked key schedule [56]. The key schedule of DES is modified such that 1 shift in round 2 and round 16 changes to 2 shifts. then, shift pattern is depicted in Figure 7.6. Sliding is based on the following theorem which is adapted from the theorem in [57].

Theorem 7.2.1 *For any key K, there exists another key K^* such that*

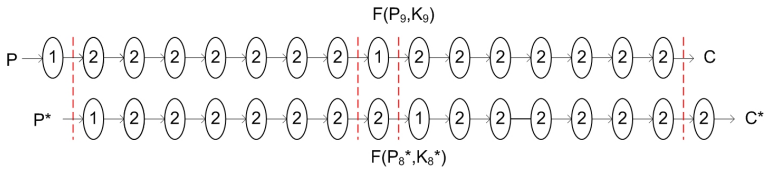

Figure 7.6: Shift pattern

$K_{i+1} = K_i^*$ *where* $i \in \{1,2,3,4,5,6,7\} \cup \{9,10,11,12,13,14,15\}$ *i.e K and* K^* *share 14 round keys.*

Proof. For the sake of simplicity, we will use the same notations introduced by Knudsen: 56 bit permuted key K is separated into two parts such that $K = (C_0, D_0)$, then the subkeys are is defined as $K_i = PC_2(C_i, D_i)$ where $C_i = LS_i(C_{i-1})$, $D_i = LS_i(D_{i-1})$ and LS_i is a circular shift to the left according to the number of positions which is given in Table 7.1. Further, define $L_{a[i]}(C_0, D_0) = (LS_{a[i]}(C_0), LS_{a[i]}(D_0))$ where $a[i]$ is the summation of all shifts up to round $i+1$, and $K_i = PC2(L_{a[i]}(K))$ and $K = (C_0, D_0)$.

Table 7.1: Circular Shifts in the Tweaked Key Schedule of DES

Round i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
LS_i	1	2	2	2	2	2	2	2	1	2	2	2	2	2	2	2
$a[i]$	1	3	5	7	9	11	13	15	16	18	20	22	24	26	28	30

Let K be given and set $K^* = L_2(K)$, then $K_2 = PC2(L_3(K)) = PC2(L_1(K^*)) = K_1^*$, $K_9 = PC2(LS_{16}(K))$ and $K_8^* = PC2(L_{15}(K^*)) = PC2(L_17(K))$, and similarly $K_{i+1} = K_i^*$ where $i \in \{1,2,3,4,5,6,7\}$. Then, the subkeys resynchronize, since $K_{10} = PC2(L_{18}(K)) = PC2(L_{16}(K^*)) = K_9^*$, then similarly $K_{i+1} = K_i^*$ where $i \in \{9,10,11,12,13,14,15,16\}$. ■

As seen in Figure 7.6, the rounds from 2 to 8 and 10 to 16 of the first encryption and the rounds from 1 to 7 and 8 to 15 of the second encryption have the same round keys. By this way, it is not possible to apply typical slide attack, because there is an unslid round in round 9 of the first encryption and in round 8 of the second encryption. However, realigning slide technique handles this situation by sliding the inharmonious round probabilistically.

Let F be the round function of DES, once $F(P, K_1) = P^*$ and $F(X_9, K_9) = F(X_8^*, K_8)$ with a probability p are satisfied, then the relation $F(C, K_{16}) = C^*$ is obtained for free. Such a pair P and P^* is a slid pair if the relations $F(P, K_1) = P^*$ and $F(C, K_16) = C^*$ are satisfied. Then, due to the Feistel structure of DES following equations obtained for a slid pair:

$$P_L^* = P_R \tag{7.10}$$

$$P_R^* = P_L \oplus F(P_R, K_1) \tag{7.11}$$

$$C_R^* = C_L \tag{7.12}$$

$$C_L^* = C_R \oplus F(C_R^*, K_{16}) \tag{7.13}$$

The crucial point is to compute the probability p to get rid of the unslid round. The probability p can be computed by using DDTs or XOR Tables of S-boxes of DES. When DDTs of S-boxes are analyzed, it is seen that among 2^6 possible input differences, the average number of output differences is 34.25 which cause zero output difference. Therefore, the probability p of $F(X_9, K_9) = F(X_8^*, K_8)$ is computed as $(\frac{34,24}{64})^8 \approx 2^{-8}$.

Note that input difference to unslid round in both encryptions is zero and nonzero input difference to S-boxes is due to the key differences. Therefore, out of 2^{56} possible key pairs about $2^{56} \cdot 2^{-8} = 2^{48}$ pairs satisfy the desired condition $F(X_9, K_9) = F(X_8^*, K_8)$.

To perform the attack, pick two sets of plaintexts each of which have $(\frac{1}{p})^{1/2} \cdot 2^{16}$ plaintexts P_i and P_j^*, respectively. Then, obtain their corresponding ciphertexts C_i and C_j^* which provide $\frac{1}{p} \cdot 2^{32}$ chosen plaintext-ciphertext pairs. Therefore, it is expected to find about $p \cdot 2^{-32} \cdot \frac{1}{p} \cdot 2^{32} = 1$ slid pair.

Since Equation 7.12 provides 32-bit filtering condition on a slid pair, there are left about $\frac{1}{p}$ pairs to examine. Then, each remaining pair provides 2^{16} candidates for K_1 and 2^{16} key candidates for K_{16} from Equations 7.12 and 7.12 respectively, in total 2^{32} key pairs (K_1, K_{16}). Since K_1 and K_{16} have 40 bits in common which provides 40-bit filtering condition, there are $2^{32} \cdot 2^{-40} = 2^{-8}$ key candidates left. Since the number of remaning pairs is $\frac{1}{p}$, the number of

suggested key pairs is $\frac{1}{p} \cdot 2^{-8}$. In this attack, we have $p = 2^{-8}$, therefore, it is expected to get 1 real key value (K_1, K_{16}).

The time complexity of the attack is $2^8 \cdot \frac{2}{16} = 2^5$ full round DES encryptions, since checking equation for one pair is equivalent to 2-round DES encryptions.

Note that in Equation 7.11, if $P_R^* \oplus P_L$ is rolled back to the f-function of DES, there are 2^2 input value to the each S-box. The reason is that S-boxes of DES are 6 bit to 4 bit and there are 2^2 input values correspond to each output value. Therefore, for each candidate slid pair, each S-box suggests 2^2 candidates for K_1 and in total there are $(2^2)^8 = 2^{16}$ candidates for K_1. In addition, in Equation 7.13, similarly there are 2^{16} candidates for K_{16}.

7.2.4 Methods for Handling Stronger Functions

As mentioned before, a typical slide attack is based on the assumption that the cipher is decomposed into identical permutation which is so weak that a few input-output pairs are enough to derive the key. But if the permutation is so strong that multiple input output pairs are needed to derive the secret key. Therefore, to cope with this situation, two sliding methods presented by Biryukov and Wagner [55] is described below:

1. **By using Differential Analysis:** Let $\alpha \rightarrow \beta$ be a differential characteristic with probability p for the permutation F. Let P and P^* be two plaintexts and $P \oplus \alpha$ and $P^* \oplus \beta$ be their related plaintexts, respectively. If we suppose that $F(P) = P^*$, then $F(P \oplus \alpha) = P^* \oplus \beta$ holds with probability p due to differential characteristic above which provides a slid pair P and P^*. By this way, for the permutation F, there are four known input-output pairs to derive the key. If a set of $3 \cdot 2^{n/2} p^{-1/2}$ plaintexts are chosen including the plaintexts P and $P \oplus \alpha$ and $P \oplus \beta$, then we have one slid pair P and P^* such that $F(P) = P^*$ and $F(P \oplus \alpha) = P^* \oplus \beta$ holds with probability p.

2. **By using multiple encryption:** If the permutation F needs N known plaintexts to derive the key K, then the following method works out to find N slid pairs. For each

plaintex P, obtain its encryption $E(P)$ and encryption of $E^2(P)$ of $E(P)$ and until the encryption $E^{2N}(P)$. Suppose if $P^* = F(E^i(P))$, then by sliding condition on ciphertexts $E(P^*) = F(E^{i+1}(P))$ holds for free. By this way, we get the relation $E^j(P^*) = F(E^{i+j}(P))$ for $j = 1, 2, \ldots, 2N-1$ which provides $2N - j$ slid pairs. If $N^{1/2} \cdot 2^{n+1/2}$ known plaintexts which constitute $N \cdot 2^n$ plaintext-ciphertext pairs are chosen, N slid pairs the existence of which are verified by the birthday paradox can be identified by using this technique.

7.3 Improved Slide Attacks

Biham et al. [58] have improved the slide attack in a way that one can find slid pairs faster than early presented techniques. In the previous techniques, existence of a slid pair is confirmed by the Birthday Paradox, but no obvious methods are provided to find a slid pair. On the contrary, this technique provides an instant method to find several slid pairs by exploiting the cycle structure of the encryption algorithm and the round function of the given cipher.

7.3.1 Improved Slide Technique

In this method, the cycle structure of the cipher is analyzed and an attack which is based on on only the cycle structures of E and F and not based on any other properties of E and F is constructed.

Let $E = F \circ F \circ \cdots \circ F = F^r$ an encryption algorithm of a block cipher which can be decomposed into r-identical permutations.

Definition 7.3.1 *Let $P : GF(2^n) \rightarrow GF(2^n)$ be a random permutation, then the cycle length of X is defined as $Cyclelenght(X) = min\{s > 0 | P^s(X) = X\}$ where $X \in GF(2^n)$.*

As mentioned in [59], the length of cycles are close to be uniformly distributed, therefore the expected values of the cycle length is $E(Cyclelenght(X)) = 2^{n-1}$. Denote the cycle length of plaintext X in F and E by $Cycle_F(X)$ and $Cycle_E(X)$ respectively and let the value of $Cycle_F(X)$ and $Cycle_E(X)$ be a and b, respectively. Since $E = F^r$, the relation $E^b(X) = X$ can

be expressed as $X = E^b(X) = F^{r \cdot b}(X)$. Since, a is the cycle length of X in the cycle of F, a divides $r \cdot b$, that is $a \mid r \cdot b$. If not so, assume the contrary that $a \nmid r \cdot b$, more explicitly $r \cdot b = a + r_1$, $0 < r_1 < a$, then we have $X = F(X) = F^{a+r_1}(X) = F^{r_1}(X)$ which contradicts to the assumption that a is the smallest integer satisfying the relation $F^a(X) = X$. Hence, $a \,|\, r \cdot b$.

Proposition 7.3.2 $b = \dfrac{a}{gcd(a, r)}$

Proof. We need to show that $b \left| \dfrac{a}{gcd(a, r)} \right.$ and $\dfrac{a}{gcd(a, r)} \,|\, b$.

($\Rightarrow$:) $\dfrac{a}{gcd(a, r)} \,|\, b$ is obvious, since $a \,|\, r \cdot b$.

($\Leftarrow$:) Consider $E^{\frac{a}{gcd(a,r)}}(X) = F^{r \cdot \frac{a}{gcd(a,r)}}(X) = (F^a(X))^{\frac{r}{gcd(a,r)}} = X$. Since b is the cycle length of X in the cycle of E, $b \left| \dfrac{a}{gcd(a, r)} \right.$. ■

If $gcd(a, r) = 1$, then by Proposition 7.3.2 we get $a = b$. By Euclid's Extended Algorithm, there exits an integer $k_1 \in \{0, 1, \ldots r - 1\}$ such that $k_1 \cdot a = -1 (mod\ r)$ or equivalently there exits k_2 such that $k_2 \cdot r = k_1 \cdot a + 1$. Then, since $F^{k_1 \cdot a + 1}(X) = (F^{k_1 \cdot a} \circ F)(X) = F(X)$ and $F^{k_1 \cdot a + 1}(X) = F^{k_2 \cdot r}(X) = E^{k_2}(X)$, it is found that $F(X) = E^{k_2}(X)$ which gives a slid pair $(X, E^{k_2}(X))$. By using multiple encryption technique [55] which is also mentioned in subsection 7.2.4, $(E^t(X), E^{k_2+t}(X))$ are slid pairs where t is an integer satisfying $1 \leq t \leq b - 1$, too.

If $gcd(a, r) = 1$, then we have seen that one can find slid pairs, instantly. However, if it is not the case, the success probability of the attack is computed as: for a random permutation the probability of $gcd(a, r) \neq 1$ is $\dfrac{r - \varphi(r)}{r}$ where φ is an Euler Phi Function. For choosing t plaintexts which are all in different cycles, the success probability of the attack is $1 - (\dfrac{r - \varphi(r)}{r})^t$.

Since expected value of cycle length for any X is $E(Cycle_F(X)) = 2^{n-1}$, the attack needs $O(2^{n-1})$ adaptively chosen plaintexts.

The algorithm for finding slid pairs is the following:

1. Take a plaintext X_0 and compute b by encrypting X_0 in succession until to get X_0 again. Note that there are no algorithms to compute a.

2. Assume that $gcd(a, r) = 1$, then compute k_2 to find slid pairs $(E^t(X), E^{k_2+t}(X))$ where $t \in \{0, \ldots, b-1\}$. The number of slid pairs is b, since b is the cycle length of X in the cycle E.

3. If the algorithm fails, then take another plaintext X_1 such that $X_1 \notin Cycle_E(X_1)$ and redo the steps 1 and 2.

In paper [58], this method is applied to 24-Round GOST.

7.3.2 Improved Slide Attack on 24-Round GOST

GOST [16] is vulnerable to this technique due to the self-similarity of its key schedule. For the detailed description of GOST, refer to Chapter 3. If 24-round of GOST is considered, the cipher can be decomposed into 3 identical permutations each of which have 8-rounds. Consider the encryption function of GOST as $E = F^3$, where F is an 8-round permutation excepting the set of subkeys $\{K_1, K_2, \ldots K_3\}$. First of all, slid pairs $((P_i, C_i), (P_j, C_j))$ such that $F(P_i) = P_j$ and $F(C_i) = C_j$ for some i and j are found, then an 8-round differential attack is applied. The crucial point of the attack is that a 8-round differential attack is applied to 8-round F permutation instead of 24-round encryption function E. For this reason, the attack considers slid pairs (P_i, P_j) as plaintext-ciphertext pairs for F permutation. More explicitly, P_j is the corresponding ciphertext of P_i, that is $F(P_i) = P_j$.

The attack is based on a 7-round differential characteristic which is independent of S-Boxes of GOST. The 7-round differential characteristic given in Table 7.2 has a probability of $\frac{15}{16} \cdot \frac{45}{64} \cdot \frac{3}{4} = 0.494$ where $A_i \in \{0, 1_x, \ldots, 8_x\}$, $B_i \in \{0, 8_x\}$ and $\overline{B_1} \in \{1, 9_x\}$.

The attack procedure is as follows:

1. By using the technique given in Section 7.3.1, find $2^{43.5}$ slid pairs (P_i, P_j) such that $F(P_i) = P_j$.

Table 7.2: A 7-Round Differential Characteristics with Probability of 0.494 for GOST

Round i	ΔX_i^L	ΔX_i^R	Probability
1	00010000	00000000	1
2	00000000	00010000	$\frac{15}{16}$
3	00010000	$A_1B_1000000$	1
4	$A_1B_1000000$	$0001A_2?B_2$	$\frac{45}{64}$
5	$0001A_2?B_2$	$A_5B_5A_3??B_300$	1
6	$A_5B_5A_3??B_300$	$???\overline{B_1}0A_7??$	$\frac{3}{4}$
7	$???\overline{B_1}0A_7??$	$?A_8B_9?????$	1
	????????	$?A_8B_9?????$	-

2. Construct a set of pairs of slid pairs such that $((P_i, P_j), (P_i^*, P_j^*))$ such that $P_i \oplus P_i^* = \Delta P$.

   From the set of $2^{43.5}$ slid pairs, it is expected to find $\frac{(2^{43.5})^2}{2} \cdot 2^{-64} = 2^{22}$ pairs of slid pairs having the difference ΔP where 2^{-64} is the probability of having the difference ΔP.

3. Pick pairs of slid pairs $((P_i, P_j), (P_i^*, P_j^*))$ such that

   - P_j has the value $A0$ in bit positions $8-15$, where $A \in \{0_x, 1_x, \ldots, F_x\}$ is a predetermined value
   - P_j^* has the value $B0$ in bit positions $8-15$, where $B \in \{0_x, 1_x, \ldots, F_x\} - \{A\}$ is a predetermined value

     Since, we have 8-bit condition on both set of ciphertexts, $2^{22} \cdot (2^{-8})^2 = 2^6$ ciphertexts remain.

     Note that determining some bits of ciphertexts can be done to decrease the data complexity of the attack. On the other hand, choosing zero value in bit positions $8-11$ in both set of ciphertexts decreases the probability of carry bit which changes the input to S_4 reduces significantly also, makes same input value to S_4. However, the attack can fail if the four key bits corresponds to S_3 are all 1's and a carry comes from the input to S_4. If it happens, the attack fails and more conditions should be imposed on ciphertexts.

4. For each remaining slid pair $((P_i, P_j), (P_i^*, P_j^*))$,

   - Guess the output of S_4 for all pairs (P_i, P_j) under the assumption that the output value of all pairs are equal

- Guess the output of S_4 for all pairs (P_i^*, P_j^*) under the assumption that the output value of all pairs are equal
- Partially decrypt ciphertexts P_j and P_j^* through S_4 and check if the difference is 0 in 4 bits (23-26) of the left half of the input
- If the difference in these 4-bits are zero increment the counter by 1 corresponds to 8-bit guesses

5. Keep the guesses whose counter is greater than 19.

The idea behind the attack is to exploit the fact that in right pairs there is a difference of 4-bits. This is done by making sure that all the 'ciphertexts' (P_j, P_j^*) are expected to have the same input to S-boxes thus the same output. This is why the outputs of S_4 are guessed.

In addition, among 2^6 slid pairs, the counter of wrong guesses is $2^6 \cdot 2^{-4} = 4$ while the counter of a right pair is $2^6 \cdot 0.494 = 31.6$. Since there are 2^8 guesses in total , the correct guessed is suggested with a probability of $1 - 2^{-7.6}$ while a wrong guessed has counter 19 with a probability of less that 2^{-22}.

The time complexity of the attack is dominated by the first step which finds slid pairs by using about 2^{63} encryptions.

CHAPTER 8

CONCLUSION

Block ciphers play crucial roles in many cryptographic real-life applications. Accordingly, day by day new block ciphers are designed to be used in such applications. In parallel to this, the security of these newly designed block ciphers should be analyzed and their reliability should be proved. Therefore, cryptanalysis of block ciphers is as important as designing secure block ciphers and plays a key part in cryptology.

In this work, we have covered cryptanalytic attacks based on related keys for block ciphers. Recently, related-key attacks have been applied to many block ciphers and gain importance in security analysis of the block ciphers. Although related-key attacks seem infeasible to be mounted at first glance and some cryptographers claim that it is not always possible to make encryptions under keys that have special properties (difference, etc.), an ideal cipher should never have such relations between plaintext-ciphertexts and keys. Moreover, related-key attacks can be effectively applied to cryptographic block cipher based hash functions. To give an example, new striking related-key attacks on full-round AES-192 and AES-256 have recently been presented by Biryukov et al. [10]. They proved in theoretically that AES-192 is not an ideal cipher and can not be used as an hash function in Davies-Meyer mode anymore.

To conclude, this work have presented cryptanalytic attacks exploiting the structure of the encryption and key scheduling algorithms of block ciphers. The aim is to combine all types of related-key attacks on block ciphers. Furthermore, this work includes our contribution which improves 12-round impossible distinguisher for XTEA by exploiting differential property of the key schedule of XTEA. The summary of the chapters is as follows. Chapter 1 includes

an introduction of block ciphers and brief descriptions of differential and linear cryptanalysis. Chapter 2 presents basic related-key attacks and its applications. Chapter 3 is devoted to related-key differential cryptanalysis and its applications. Chapters 4, 5 and 6 are on the extensions of related-key differential cryptanalysis, namely related-key boomerang, related-key amplified boomerang, related-key rectangle, related-key impossible differential, related-key impossible boomerang and related-key differential-linear attacks. Chapter 7 gives the detailed description of slide attack and its improvements.

REFERENCES

[1] E. Biham and A. Shamir, *Differential Cryptanalysis of DES-like Cryptosystems*, Advances in Cryptology, Proceedings of CRYPTO'90, 10th Annual International Cryptology Conference, Lecture Notes in Computer Science, volume 537, Springer, 1991.

[2] E. Biham and A. Shamir, *Differential Cryptanalysis of Data Encryption Standard*, Springer-Verlag, 1993.

[3] E. Biham and A. Shamir, *Differential Cryptanalysis of the full 16-round DES*, Advances in Cryptology, Proceedings of CRYPTO'92, The 12th Annual International Cryptology Conference, Lecture Notes in Computer Science, volume 740, pages 487-496, Springer-Verlag, 1993.

[4] M. Matsui and A. Yamagishi, *A New Method for Known Plaintext Attack of FEAL cipher*, Advances in Cryptology, Proceedings of EUROCRYPT'92, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes in Computer Science, volume 658, pages 81-91, Springer-Verlag, 1993.

[5] M. Matsui, *Linear Cryptanalysis Method for DES cipher*, Advances in Cryptology, Proceedings of EUROCRYPT'93, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes in Computer Science, volume 765, pages 386-397, Springer-Verlag, 1994.

[6] R. Winternitz and M. E. Hellman, *Chosen-Key Attacks on a Block Cipher*, Cryptologia, volume 11, pages 16-20, 1987.

[7] L. R. Knudsen, *Cryptanalysis of LOKI*, Advances in Cryptology, Proceedings of ASIACRYPT 1992, volume 739, Lecture Notes in Computer Science, pages 22-35, Springer-Verlag, 1992.

[8] E. Biham, *New Types of Cryptanalytic Attacks Using Related-Key*, Advances in Cryptology, EUROCRYPT'93, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes on Computer Science, volume 765, pages 229-246, Springer-Verlag, 1994.

[9] L. Brown, J. Pieprzyk and J. Seberry, *LOKI- A Cryptographic Primitive for Authentication and Secrecy Applications*, Proceedings of AUSCRYPT'90, International Conference on Cryptology in Australia, Advances in Cryptology, Lecture Notes in Computer Science, volume 453, pages 229-236, 1990.

[10] A. Biryukov and D. Khovratovich, *Related-Key Cryptanalysis of the Full AES-192 and AES-256*, 2009.

[11] E. Biham, O. Dunkelman, and N. Keller, *A Simple Related-Key Attack on the full SHACAL-1*, CTRSA 2007, Topics in Cryptography, Lecture Notes in Computer Science, volume 4377, pages 20-30, Springer, 2007.

[12] H. Handschuh and D. Naccache, *SHACAL*, Preproceeding of NESSIE first workshop, Leuven, 2000.

[13] J. Kelsey, B. Schneier and D. Wagner, *Key Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES*, Proceedings of CRYPTO'96, The 16th Annual International Cryptology Conference, Advances in Cryptology, pages 7-23, Springer-Verlag, 1996.

[14] Y. Ko, S. Hong, W. Lee, S. Lee and J.-S. Kang ,*Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round GOST*, FSE'04, Lecture Notes in Computer Science, volume 3017, pages 299-316, Springer Berlin / Heidelberg, 2004.

[15] M. Blunden and A. Escott, *Related-Key Attacks on Reduced Round KASUMI*, Proceedings of FSE'01, 8th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 2355, pages 277-285, Springer-Verlag, 2002.

[16] GOST, Gosudarstvennyi Standard 28147-89, *Cryptographic Protection for Data Processing Systems*, Government Committee of the USSR for Standards, 1989.

[17] H.Seki and T.Kaneko, *Differential Cryptanalysis of Reduced Rounds of GOST*,SAC'00 , The 7th Annual Workshop on Selected Areas in Cryptography, Lecture Notes in Computer Science, pages 315-323, volume 2012, Springer-Verlag, 2001.

[18] The 3rd Generation Partnership Project, Technical Specification Group Services and System Aspects, 3GSecurity, *Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification,* V.3.1.1, 2001.

[19] M. Matsui, *Block Encryption Algorithm MISTY*, Proceedings of FSE'97, The 4th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 1267, pages 6474, Springer-Verlag, 1997.

[20] E. Biham, O. Dunkelman and N. Keller, *A Related-Key Rectangle Attack on the Full KASUMI*, ASIACRYPT'05, Annual International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science, volume 3788, pages 443461, Springer, 2005.

[21] R. Knudsen, *DEAL - A 128-Bit Block Cipher*. Technical report, Department of Informatics, University of Bergen, Norway, 1998.

[22] E. Biham, A. Biryukov and A. Shamir, *Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials*, Advances in Cryptology, Proceedings of EUROCRYPT 1999, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes in Computer Science, volume 1592, pages 12-13, Springer Verlag, 1999.

[23] E. Biham, A. Biryukov and A. Shamir, *Miss in the Middle Attacks on IDEA and Khufu*, Proceedings of FSE'99, The 6th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 1636, pages 124-138, Springer-Verlag, 1999.

[24] O. Özen, K. Varıcı, C. Tezcan and Ç. Kocair, *Lightweight Block Ciphers Revisited: Cryptanalysis of Reduced Round PRESENT and HIGHT*, to appear in Information Security and Privacy, ACISP 200914th Australasian Conference, , Lecture Notes in Computer Science, Springer-Verlag, 2009.

[25] J. Daemen and V. Rijmen, *AES Proposal: Rijndael*, Second Version, AES submission.

[26] G. Jakimoski and Y. Desmedt, *Related-Key Differential Cryptanalysis of 192-bit Key AES Variants*, SAC'03, The 10th Annual Workshop on Selected Areas in Cryptography, Lecture Notes in Computer Science, volume 3006, pages 208-221, Springer-Verlag, 2004.

[27] E. Biham, O. Dunkelman and N. Keller, *Related-Key Impossible Differential Attacks on 8-round AES-192*, CT-RSA'06, The Cryptographers' Track at the RSA Conference, Lecture Notes in Computer Science, volume 3860, pages 39-49, Springer-Verlag, 2006.

[28] W. Zhang, W. Wu, L. Zhang and D. Feng, *Improved Related-Key Impossible Differential Attacks on Reduced-Round AES-192*, SAC'03, The 13th Annual Workshop on Selected Areas in Cryptography, Lecture Notes in Computer Science, volume 4356, pages 15-27, Springer-Berlin, 2007.

[29] W. Zhang, W. Wu and L. Zhang, *Related-Key Impossible Differential Attacks on Reduced-Round AES-256*.

[30] D. Hong, J. Sung, S. Hong, J, Lim, S. Lee, B. Koo, C. Lee, D. Chang, j. Lee, K. Jeong, H. Kim, J. Kim and S. Chee, *HIGHT: A New Block Cipher Suitable for Low-Resource Device*, CHES'06, The 8th International Workshop on Cryptographic Hardware and Embedded Systems pages, Lecture Notes in Computer Science, volume 4249, pages 46-59, 2006.

[31] J. Lu, *Cryptanalysis of Block Ciphers*, Royal Holloway, University of London, 2008.

[32] D. Wheeler and R. Needham, *TEA, A Tiny Encryption Algorithm*, Proceedings of FSE'98, The 9th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 1372, pages 97-110, Springer-Verlag, 1998.

[33] D. Wheeler and R. Needham, *TEA Extensions*, Technical Report, University of Cambridge, Cambridge, UK, 1997.

[34] S. Hong, Y. Ko, D. Chang, W. Lee and S. Lee, *Differential Cryptanalysis of TEA and XTEA*, Proceedings of ICISC'03, The 7th Annual International Conference on Information Security and Cryptology, Lecture Notes in Computer Science, volume 2791, pages 402-417, Springer-Verlag, 2003.

[35] E. Lee, D.Hong, D. Chang, S. Hong and J. Lim, *A Weak Key Class of XTEA for a Related-Key Rectangle Attack*, Proceedings of Vietcrypt'06, International Conference on Cryptology in Vietnam , Lecture Notes in Computer Science, volume 4341, pages 286297, Springer-Verlag, 2006.

[36] D. Moon, K. Hwang, W. Lee, S. Lee, and J. Lim, *Impossible Differential Cryptanalysis of Reduced Round XTEA and TEA*, Proceedings of FSE'02, The 9th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 2365, pages 49-60, Springer-Verlag, 2002.

[37] E. Biham, O. Dunkelman, and N. Keller, *A Related-Key Boomerang and Rectangle Attacks*, Advances in Cryptology, Proceedings of EUROCRYPT'05, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes in Computer Science, volume 3494, pages 507-525, Springer-Verlag, 2005.

[38] D. Wagner, *The Boomerang Attack*, Proceedings of FSE'99, The 10th International Fast Software Encryption Workshop Lecture Notes in Computer Science, volume 1636, pages 156170, Springer-Verlag, 1999.

[39] J. Kim, G. Kim, S. Hong, S. Lee and D.Hong, *The Related-Key Rectangle Attack - Application to SHACAL-1*, Proceedings of ICICS'04, International Conference on Information Security and Privacy, Lecture Notes in Computer Science, volume 3108, pages 123-136, Springer Verlag, 2004.

[40] S. Hong, J. Kim, S. Lee, and B. Preneel, *Related-Key Rectangle Attacks on Reduced Versions of SHACAL-1 and AES-192*, Proceedings of FSE'05, The 14th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 3557, pages 368-383, Springer-Verlag, 2005.

[41] X. Lai and J. L.Massey, *A Proposal for a New Block Cipher Encryption Standard*, Proceeding of EUROCRYPT 90, Workshop on the Theory and Application of of Cryptographic Techniques , Lecture Notes in Computer Science, volume 473, pages 389404, Springer-Verlag,1991.

[42] K. Jeong, C. Lee, J. Sung, S. Hong and J. Lim, *Related-key Amplified Boomerang Attack on the Full-Round Eagle-64 and Eagle-128*, Information Security and Privacy, ACISP 2007, 12th Australasian Conference, Lecture Notes in Computer Science, volume 4586, pages 143-157, Springer-Verlag, 2007.

[43] E. Biham, O.Dunkelman and N. Keller, *The Rectangle Attack Rectangling the Serpent*, Proceedings of EUROCRYPT'01, International Conference on the Theory and Application of Cryptographic Techniques, Lecture Notes in Computer Science, volume 2045, pages 340357, Springer-Verlag, 2001.

[44] J. Kim, S. Hong and B. Preneel, *Related-Key Rectangle Attacks on Reduced AES-192 and AES-256*, Proceedings of FSE'07, The 14th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 4593, pages 225-241, Springer-Verlag, 2007.

[45] E. Biham, O. Dunkelman, and N. Keller, *Rectangle Attack on 49-round SHACAL-1*, Proceedings of FSE'03, The 10th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, pages 22-35, Springer, 2003.

[46] O. Dunkelman, N. Keller and J. Kim, *Related-Key Rectangle Attack on the Full SHACAL-1*, Proceedings of SAC'06, Selected Areas in Cryptography, Springer, 2006.

[47] X. Wang, L. Yin, H. Yu, *Finding Collisions in the Full SHA-1*, Advances in Cryptology, Proceedings of CRYPTO'05, The 25th Annual International Cryptology Conference, Lecture Notes in Computer Science, volume 3621, pages 17-36, 2005.

[48] US National Bureau of Standards, *Secure Hash Standard*, Federal Information Processing Standards Publications No. 180-2, 2002.

[49] S. Langford and M. Hellman, *Differential-Linear Cryptanalysis*, Advances in Cryptology, Preecedings of CRYPTO'94, The 14th Annual International Cryptology Conference, Lecture Notes in Computer Science, volume 839, pages 17-25, Springer-Verlag, 1994.

[50] E. Biham, O. Dunkelman and N. Keller, *Enhancing Differential-Linear Cryptanalysis*, Advances in Cryptology, Preecedings of ASIACRYPT'02, The 8th Annual International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science, volume 2501, pages 254-266, Springer-Verlag, 2002.

[51] W. Zhang, L. Zhang, W. Wu and D. Feng, *Related-Key Differential-Linear Attacks on Reduced AES-192*, INDOCRYPT 2007, The 8th International Cryptology Conference in India, Lecture Notes in Computer Science, volume 4859, pages 73-85, Springer-Heidelberg 2007.

[52] A. Biryukov and D. Wagner, *Slide Attacks*, Advances in Cryptology, Proceedings of FSE'99, The 6th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 1636, pages 245-259, Springer-Verlag, 1999.

[53] M. Ciet, G. Piret and J.-J. Quisquarter, *Related-Key and Slide Attacks: Analysis, Connections, and Improvements* , unpublished, 2002.

[54] L. R. Knudsen, *Cryptanalysis of LOKI91*, Advances in Cryptology, ASIACRYPT'92, Annual International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science, volume 718, pages 196-208, Springer-Verlag, 1993.

[55] A. Biryukov and D. Wagner, *Advanced Slide Attacks*, Advances in Cryptology, Proceedings of EUROCRYPT'00, International Conference on the Theory and Application of Cryptographic Techniques Lecture Notes in Computer Science, volume 1807, pages 586-606, Springer-Verlag, 2000.

[56] R. C. W. Phan, *Advanced Slide Attacks Revisited: Realigning Slide on DES*, Progress in Cryptology, Mycrypt 2005, 1st International Conference on Cryptology in Malaysia, Lecture Notes in Computer Science, volume 3715, pages 263-276, Springer, 2005.

[57] L. R. Knudsen, *New Potentially Weak Keys for DES and LOKI*, Proceedings of EUROCRYPT'94, International Conference on the Theory and Application of Cryptographic Techniques, volume 950, Springer Verlag, 1994.

[58] E. Biham, O. Dunkelman and N. Keller, *Improved Slide Attacks* Proceedings of FSE'07, 14th International Fast Software Encryption Workshop, Lecture Notes in Computer Science, volume 4593, pages 153-166, Springer-Verlag, 2007.

[59] D. W. Davies and G. I. P. Parkin, *The Average Cycle Size of the K Stream in Output Feedback Encipherment (Abtract)*, Advances in Cryptology, Proceedings of CRYPTO'82 , 2nd Annual International Cryptology Conference, Lecture Notes in Computer Science, pages 97-98, 1982.

[60] M. Matsui, *The First Experimental Cryptanalysis of Data Encryption Standard*, Advances in Cryptology, Proceedings of CRYPTO'93, 13th Annual International Cryptology Conference, Lecture Notes in Computer Science, volume 839, pages 1-11, Springer-Verlag, 1994.

[61] O. Dunkelman, *Techniques for Cryptanalysis of Block Ciphers*, PhD Thesis, Computer Science Department, Tecnion, 2006.

[62] S. Furuya, *Slide Attacks with a Known-Plaintext Cryptanalysis* Proceedings of Information and Communication Security 2001, Lecture Notes in Computer Science, volume 2288, pages 214-225, Springer-Verlag, 2000.