BASIC CRYPTANALYSIS METHODS ON BLOCK CIPHERS

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

BY

DİLEK ÇELİK

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

MAY 2010

Approval of the thesis:

BASIC CRYPTANALYSIS METHODS ON BLOCK CIPHERS

submitted by **DİLEK ÇELİK** 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. Ferruh ÖZBUDAK
Department of Mathematics, METU ________________

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

Assist. Prof. Dr. Zülfükar SAYGI
Department of Mathematics, TOBB ETU ________________

Dr. Muhiddin UĞUZ
Department of Mathematics, METU ________________

Dr. Murat CENK
Department of Cryptography, 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:	DİLEK ÇELİK
Signature :	

ABSTRACT

BASIC CRYPTANALYSIS METHODS ON BLOCK CIPHERS

Çelik, Dilek
M.S., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Ali DOĞANAKSOY

May 2010, 119 pages

Differential cryptanalysis and linear cryptanalysis are the first significant methods used to attack on block ciphers. These concepts compose the keystones for most of the attacks in recent years. Also, while designing a cipher, these attacks should be taken into consideration and the cipher should be created as secure against them.

Although differential cryptanalysis and linear cryptanalysis are still important, they started to be inefficient due to the improvements in the technology. So, these attacks are extended. For instance, higher order differential cryptanalysis, truncated differential cryptanalysis, generalized linear cryptanalysis, partitioning linear cryptanalysis, linear cryptanalysis using multiple linear approximations are introduced as the extended versions of these attacks. There exists significant applications of these extended attacks.

Algebraic attack is a method of cryptanalysis that consists of obtaining a representation of the cipher as a system of equations and then, solving this system. Up to today, just a few attacks that are practically possible to mount are presented. However, due to the fact that algebraic cryptanalysis requires only a handful of known plaintexts to perform, it is a promising and significant attack.

This thesis is a survey covering all the methods of attacks described above. Illustrations and summaries of some important papers including these cryptanalysis techniques are given.

Keywords: Block Ciphers, Differential Cryptanalysis, Linear Cryptanalysis, Algebraic Attack.

ÖZ

BLOK ŞİFRELER ÜZERİNE UYGULANAN TEMEL KRİPTANALİZ METOTLARI

Çelik, Dilek
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali DOĞANAKSOY

Mayıs 2010, 119 sayfa

Diferansiyel kriptanaliz ve lineer kriptanaliz blok şifreler üzerine uygulanmış olan ilk önemli ataklardır. Bu kriptanaliz metotları günümüzde uygulanan çoğu atağın temellerini oluşturmaktadır. Ayrıca, bir şifre dizayn ederken bu ataklar göz önünde bulundurulmalı ve şifreler bu ataklara dayanıklı olarak hazırlanmalıdır.

Diferansiyel kriptanaliz ve lineer kriptanaliz hala önemli ataklar olmalarına rağmen, teknolojinin gelişmesiyle birlikte etkilerini yitirmeye başlamışlardır ve bu yüzden geliştirilmişlerdir. Bunlara örnek olarak, yüksek dereceli diferansiyel kriptanaliz (higher order differential cryptanalysis), kesik diferansiyel kriptanaliz (truncated differential cryptanalysis), genelleştirilmiş lineer kriptanaliz (generalized linear cryptanalysis), bölmeli lineer kriptanaliz (partitioning linear cryptanalysis), çoklu denklemler kullanılarak yapılan lineer kriptanaliz (linear cryptanalysis using multiple linear approximations) verilebilir. Bu geliştirilmiş versiyonların önemli uygulamaları bulunmaktadır.

Cebirsel atak bir şifreyi denklemler sistemi olarak ifade ettikten sonra, bu sistemi çözmeye dayanan bir kriptanaliz yöntemidir. Şimdiye kadar bu metot ile pratik olarak uygulanabilen atak sayısı çok azdır. Fakat, gelecek vaat eden önemli bir kriptanaliz yöntemidir çünkü atağı uygulamak için az miktarda düz metin ve onların şifrelenmiş metinleri gereklidir.

Bu tez yukarıda değinilmiş olan bütün atakları kapsayan bir araştırmadır. Bu kriptanaliz yöntemlerini içeren bazı önemli makalelerin özetleri verilmiş ve örneklendirilmiştir.

Anahtar Kelimeler: Blok Şifreler, Diferansiyel Kriptanaliz, Lineer Kriptanaliz, Cebirsel Ataklar.

To my parents and Vakkas

ACKNOWLEDGMENTS

Firstly, I deeply thank my supervisor Assoc. Prof. Dr. Ali DOĞANAKSOY. His help, guidance, stimulating suggestions, encouragement and motivation are very important for me throughout this thesis period.

I would like to give my special thanks to my husband Vakkas and my parents for their love, endless support, caring and helping they provided me through my entire life.

I am grateful to my collegues Aslı BAY, İ.Firuze KARAMAN and Neşe ÖZTOP for their close friendship. They have been great collaborators and supporters.

It is a great pleasure to thank also Onur KOÇAK, Fatih SULAK, Kerem VARICI, and Onur ÖZEN for assisting me in many different ways.

I would like to thank everyone with whom I have worked at Graduate School of Natural and Applied Sciences and Mathematics Department for their understanding during this period.

TABLE OF CONTENTS

CHAPTERS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

From very long time ago until today, people always needed to hide their secrets. To fulfill this need, they interested in the art of science, cryptography. The history of cryptography is long and goes back at least 4,000 years to the Egyptians, who used hieroglyphic codes for inscription on tombs [82].

Together with the developments in technology, the ciphering methods are also improved. The designers try to create their ciphers as secure enough that the attackers cannot break even with the knowledge of everything about the system. In the 19^{th} century, Auguste Kerckhoffs's stated a desiderata, and after that ciphers are designed according to this principle.

Kerckhoffs's Principle

1. If the system is not theoretically unbreakable, it should be unbreakable in practice.
2. Compromise of the system details should not give any trouble to the correspondents.
3. The key must be communicable, rememberable without notes and easily changed at the will of the correspondents.
4. The cipher should be transmissible by telegraph.
5. The encryption apparatus should be portable, and its usage and function must not require the concourse of several people, it should be operable by a single person.
6. The system should be easy to use, should require no knowledge of a long list of rules or mental strain.

Today's modern ciphers are designed by taking the Kerckhoffs's Principle into consideration. They are usually classified as two large groups, namely the stream ciphers and the block ciphers. In this thesis, the cryptanalysis of block ciphers is studied. So, we will just give some information about the block ciphers and types of cryptanalytic attacks.

1.1 Block Ciphers

Symmetric-key algorithms are a class of algorithms for cryptography such that the encryption key of the algorithm is the same with the decryption key or there is a simple transformation to go between the two keys. Block ciphers can be either symmetric-key or public-key. We will focus only on the symmetric-key block ciphers.

Definition 1.1.1 *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. Formally, it is a function from* $\{0,1\}^n \times \{0,1\}^k$ *to* $\{0,1\}^n$ *such that for each key, K in the key space, the encryption function* $E(P,K)$ *is an invertible mapping where P is the plaintext.*

Block ciphers divide the plaintext into blocks of symbols to use a specially constructed function which mixes the block of plaintext with the secret key to produce the ciphertext [81]. We can classify the structure of block ciphers into two parts: substitution-permutation network (SPN) and Feistel network.

Definition 1.1.2 *A substitution-permutation network (SPN) is a cipher combining two or more transformations in a manner intending that the resulting cipher is more secure than the individual components. It is composed of a number of stages each involving substitutions and permutations [83]. An illustration is given in figure 1.1.*

In SPN, the substitution layer provides the confusion while the permutation layer provides the diffusion.

Definition 1.1.3 *Let* L_0 *and* R_0 *be n bit blocks. A Feistel cipher is an iterated cipher mapping a* $2n$ *bit plaintext* (L_0, R_0) *to a ciphertext* (L_k, R_k)*, through a k round process where* $k \geq 1$. *For* $1 \leq i \leq k$ *, round i maps* (L_{i-1}, R_{i-1}) *to* (L_i, R_i) *using* K_i *according to the following rule:*

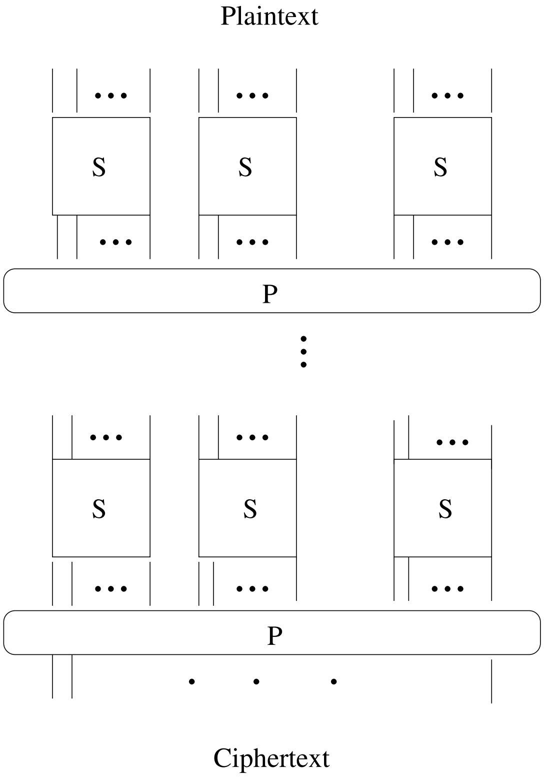


Figure 1.1: An Example of a Substitution-Permutation Network

$$L_i = R_{i-1} \text{ and } R_i = L_{i-1} \oplus F(R_{i-1}, K_i),$$

where each subkey K_i is derived from the cipher key, K [83]. DES and FEAL can be given as examples of Feistel based algorithm. An illustration for a Feistel cipher is given in figure 1.2 Also, generalized Feistels can be constructed by the dividing the plaintext into into quarters, octets, etc. at the beginning instead of dividing into two as in Feistel network. CAST family of ciphers and SkipJack can be given as examples [81]. An illustration is given in figure 1.3.

Feistel structured ciphers has an advantage over SPN ciphers that the implementation process is nearly halved for a Feistel cipher due to the similarity of the encryption and decryption operations. These operations can sometimes be even identical. In this case, one only needs to reverse the key schedule for decryption.

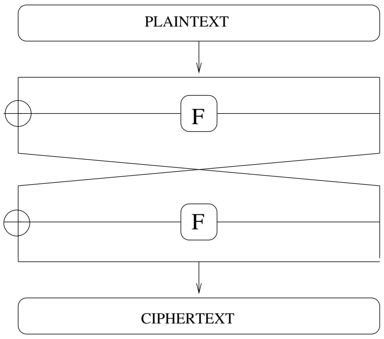


Figure 1.2: A Feistel Cipher

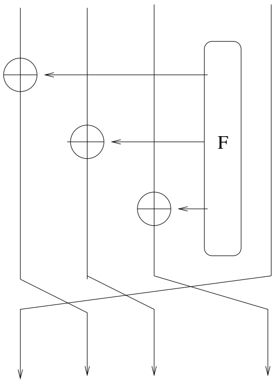


Figure 1.3: Generalized Feistel Cipher

1.2 Classification of Cryptanalytic Attacks

Breaking a cipher has different meanings that changes according to the type of the attacks. For instance, a cipher is said to be broken if we get the plaintext from the ciphertext without having a key or if the secret key is found or if the cipher can be distinguished from a random permutation.

Some of the significant cryptanalytic techniques are given below.

Ciphertext-Only Attack

In this type of attack, the attacker tries to find the decryption key or the plaintext by only having the knowledge of the ciphertext. If a cipher can be broken by a ciphertext-only attack, it can be seen as a completely insecure cipher because everybody has an access to a ciphertext.

Known-Plaintext Attack

In this type of attack, the attacker is assumed to know a number of plaintexts with their corresponding ciphertexts. Using these plaintext-ciphertext pairs it is aimed to find the key or unknown plaintexts-ciphertext pairs. Linear cryptanalysis is an example for a known-plaintext attack.

Chosen-Plaintext Attack

In this type of attack, the attacker is assumed to have an access to a number of plaintexts which he chooses and their corresponding ciphertexts. That is, the attacker has the encryption of a plaintext of his own choice. This can be possible if the attacker directly has the key or he sends the chosen plaintext to the owner of the secret key and then, an eavesdropper to the transmission of this text in encrypted form to a third party [81]. Differential cryptanalysis is an example for a chosen plaintext attack.

Chosen-Ciphertext Attack

In this type of attack, the attacker is assumed to have the corresponding plaintext of the ciphertext which he chooses.

Adaptive-Chosen Plaintext or Ciphertext Attack

In this type of attack, similar to the chosen ciphertext attack, the attacker is assumed to select the ciphertext and then, can obtain the corresponding plaintext but this time the choice of the ciphertext may depend on the plaintext received from previous requests. That is, the ciphertexts may be chosen adaptively before and after a challenge ciphertext is given to the attacker, with only the agreement that the challenge ciphertext may not itself be queried.

Related-Key Attack

In this type of attack, the attacker benefits from the relations between keys in two different encryptions. He uses these relations while attacking with one of the types explained above. For this type of attack, the key schedule of the cipher should have a theoretical weakness.

A cipher is said to be broken in practice, if the attacker obtains the concrete result in his hands. To illustrate, he finds the key exactly instead of showing the way about how to find it. In some cases, today's technology cannot be adequate to break the cipher practically but the attacker shows the way of breaking it and says if the circumstances were satisfying, it could be broken with a suitable complexity. This time the cipher is said to be broken theoretically.

It is possible to build systems that cannot be broken in practice. However, we still can never be sure that a cipher is secure or not because it can be broken theoretically. At least, one can try all possible keys in sequence one by one to find the right key. This method is called exhaustive search. There is only one known cipher that is completely secure, namely the "one-time pad". In this cipher, each letter is transposed to another letter a random distance away. That is the plaintext is just XORed with the key. The important point here is the length of the key equals to the length of the plaintext and the key is used only once.

1.3 Outline and Aim of the Thesis

The aim of this thesis is to make a survey on differential cryptanalysis and its variants, linear cryptanalysis and its variants, and algebraic cryptanalysis. Moreover, this thesis is prepared not only to show a way about how to study and to teach basic cryptanalysis methods to a beginner but also to improve attackers' ability and knowledge about cryptanalysis.

Also, this thesis is the first part of the survey that is made on block cipher cryptanalysis. The second part covers the combined attacks on block ciphers [80] and the last part studies the related-key attacks on block ciphers [79].

The thesis is organised as follows. In this chapter, we give some information about the block ciphers and cryptanalytic attacks. In Chapter 2, differential cryptanalysis is introduced by mounting an instructive attack on a very simple cipher which is defined in Appendix, namely the "sample cipher". Also, detailed explanations are given for three significant published differential attacks mounted on DES and Feal-8 [2, 5, 11]. Then, two generalizations of differential cryptanalysis, namely, the higher order differential cryptanalysis and truncated differential cryptanalysis are explained and brief applications of them are given. In Chapter 3, further published illustrations for differential cryptanalysis are reported. In Chapter 4, linear cryptanalysis is introduced by mounting an instructive attack on sample cipher. Moreover, previously published linear attacks that are mounted on DES and RC5 [52, 53, 54] are explained. Then, extended versions of basic linear cryptanalysis, namely generalized linear cryptanalysis, linear cryptanalysis using multiple approximations and partitioning cryptanalysis will be studied. In Chapter 5, some meausures of security against differential cryptanalysis, linear cryptanalysis and their variants for block ciphers is given. In Chapter 6, algebraic cryptanalysis is studied.

CHAPTER 2

DIFFERENTIAL CRYPTANALYSIS

2.1 Introduction to Differential Cryptanalysis

Differential cryptanalysis is a chosen plaintext attack invented by researchers Eli Biham and Adi Shamir in 1990 for breaking certain classes of cryptosystems. The attack depends on the observation of the effect of particular differences of plaintext pairs on the differences of the corresponding ciphertext pairs [2]. The attacker determines a particular plaintext difference ΔP, then for that given difference tries to exploit an occurance of an output difference of a certain round with high probability. Since it is a chosen plaintext attack, the corresponding ciphertexts of the plaintexts are known. Using these ciphertexts, one can distinguish a known cipher from a random permutation, or attack to a number of rounds to determine the subkeys of these rounds.

One advantage of differential cryptanalysis is that, it uses the differences of pairs which are independent from the subkeys of the rounds. For instance, assume D_1 and D_2 are two datum, K is the round subkey and $*$ is the operation combining data with the round subkey, the difference between the two datum is chosen as

$$\Delta(D_1, D_2) = D_1 * D_2^{-1}.$$

It can be easily seen the key effect is discarded for the differences after the key addition:

$$\Delta(D_1 * K, D_2 * K) = D_1 * K * K^{-1} * D_2^{-1} = \Delta(D_1, D_2).$$

In most of the ciphers, the operation $*$ is chosen as the XOR operation. For each given input difference of the S-box, the output difference is not exactly known. In other words, given input difference of the S-box, say ΔX, an output difference ΔY occurs with a probability. That is, let Y and $Y^{'}$ be the output values of the S-box corresponding to the input values X, $X^{'}$ respectively, where $\Delta X = X \oplus X^{'}$. For another input values, difference of which satisfying ΔX, there is no guarantee that the output difference of the S-box corresponding to these new pair to be equal $\Delta Y = Y \oplus Y'$ because for each pair of input with difference ΔX, one gets different output pairs with various differences after the substitution.

For differential cryptanalysis, one chooses a particular plaintext difference ΔX, aims to find a scenario such that, a particular output difference of a round occurs with a very high probability. This can be achieved by determining difference pairs of one round with high probability and then, creating a scenario by combining the rounds logically. $(\Delta X, \Delta Y)$ is referred to be a difference pair, where ΔX and ΔY are input difference and output difference of a specific S-box respectively.

2.1.1 Difference Distribution Tables

For differential cryptanalysis, it is aimed to find difference pairs with high probabilities. For this purpose, difference distribution tables, denoted by DDT, are constructed by considering the input and output values of S-boxes. This table, also called as the XOR table, shows the distribution of all input differences and output differences of each possible input-output pairs of an S-box [2]. Each entry of DDT represents the number of difference pairs with the corresponding input and output difference. So, it is an important tool for the attack.

Note that, DDT can be created both by the help of a computer or manually. For illustration, DDT of the sample cipher will be constructed manually, since the number of input-output bits of the sample cipher are few. Now, let us start with the construction of the row of the DDT corresponding to the input difference $\Delta X = 0011$. It is aimed to find the number of occurances of each output difference when the particular input difference of the S-box is $\Delta X = 0011$. For each input value of the S-box, one can determine a second input value such that the XOR of them equals to 0011. To illustrate, let X=0000, then the second input, say $\hat{X}$, is $X \oplus \Delta X = 0000 \oplus 0011 = 0011$. The outputs corresponding to $X = 0000$ and $\hat{X} = 0011$ are $Y = 1010$ and $\hat{Y} = 0101$, respectively. Then, the output difference is $\Delta Y = Y \oplus \hat{Y} = 1010 \oplus 0101 = 1111$. Applying the same procedure for all the other values of X, one can construct table 2.1.

As it is seen from table 2.1, for ΔY values 1111 appears 4 times, 1011 appears 2 times, 0011

Table 2.1: Sample Difference Pairs of the S-box

X	Y	$\hat{X} = X \oplus \Delta X$	$\hat{Y}$	$\Delta Y = Y \oplus \hat{Y}$
0000	1010	0011	0101	1111
0001	0011	0010	1000	1011
0010	1000	0001	0011	1011
0011	0101	0000	1010	1111
0100	1100	0111	1111	0011
0101	0000	0110	0010	0010
0110	0010	0101	0000	0010
0111	1111	0100	1100	0011
1000	0110	1011	0100	0010
1001	1110	1010	0001	1111
1010	0001	1001	1110	1111
1011	0100	1000	0110	0010
1100	1001	1111	1011	0010
1101	0111	1110	1101	1010
1110	1101	1101	0111	1010
1111	1011	1100	1001	0010

appears 2 times, 0010 appears 6 times, and 1010 appears 2 times. So, when $\Delta X = 0011$ is given as the input difference, the number of occurances of the output differences, ΔY is given in table 2.2.

Table 2.2: Occurances of ΔY when $\Delta X = 0011$

ΔY	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Occurance	0	0	6	2	0	0	0	0	0	0	2	2	0	0	0	4

Table 2.2, composes the row corresponding to $\Delta X = 3$ of the DDT of the sample cipher. The whole table, namely table 2.3 can be constructed by determining all the rows in a similar manner.

In table 2.3, each row represents a particular input difference while each column presents a particular output difference. Each element of the table indicates the number of occurances of the corresponding output difference, when a particular input difference is given. Note that, when a particular input difference is given, the probability of the apperance of a particular output difference can be computed by simply dividing the corresponding number on the table by 16 because ΔY can take 16 different values. To illustrate, from table 2.3, one can see that for the input difference $\Delta X = 0001$, the number of occurances of $\Delta Y = 1101$ is 4. Then, the probability of the apperance of ΔY is 4/16=1/4. Note that, for an ideally randomized cipher,

Table 2.3: Difference Distribution Table

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	2	2	0	2	2	0	0	2	4	2	0
2	0	0	2	0	2	0	2	2	0	0	2	0	2	0	2	2
3	0	0	6	2	0	0	0	0	0	0	2	2	0	0	0	4
4	0	0	0	2	0	0	2	0	0	2	4	0	2	0	0	4
5	0	2	0	0	0	0	0	6	0	2	4	0	0	0	0	2
6	0	0	0	2	2	4	0	0	4	0	0	2	2	0	0	0
7	0	2	0	2	0	2	2	0	2	2	0	0	0	4	0	0
8	0	2	0	0	2	2	0	2	0	2	0	0	2	2	0	2
9	0	0	2	0	4	2	0	0	0	4	0	2	2	0	0	0
A	0	2	0	0	0	0	0	2	2	0	0	8	0	0	2	0
B	0	0	2	2	0	2	4	2	0	0	0	0	0	2	2	0
C	0	0	0	4	2	2	0	0	0	0	2	2	0	0	4	0
D	0	0	2	2	0	0	4	0	2	0	2	0	0	2	2	0
E	0	4	2	0	4	0	0	2	2	0	0	0	0	2	0	0
F	0	4	0	0	0	0	0	0	2	2	0	0	4	0	2	2

occurance of each output difference should be 1 for a given particular input difference, but this is impossible due to some basic properties of DDT [4].

Some Characteristics of DDT

1. Each entry on the DDT is even. This is due to the reason that $X \oplus X^{'} = X^{'} \oplus X = \Delta X$ where X and $X^{'}$ are given input values. That is, the same output difference occurs for both of the ordered pairs $(X, X^{'})$ and $(X^{'}, X)$.

2. The entries of the first row of DDT are all zero, except the first entry. The first row represents the difference of the identical inputs, namely the zero difference. The identical inputs are mapped to the same output because S-box is a mapping. So, all the output differences are zero differences, when the particular input difference is zero.

3. If the S-box of a cipher is one to one, then the elements of the first column of DDT are all zero except the first entry. Having zero output difference means the two output value are the same. Since the mapping is one to one, the inputs corresponding to the same outputs must be identical.

4. The sum of the entries in each row equals to 2^n where n is the number of output bits. This is due to the reason that for a given input difference, the sum of probabilities of the occurances of all output differences should be equal to 1. For instance, the sum of

entries in rows of the sample cipher equals $16 = 2^4$. So, when an input difference is determined, the probability of appearance of an output difference is 16/16=1.

5. The sum of the entries in each column equals to 2^n where n is the number of input bits.

2.1.2 Finding One Round Differential Characteristic

Differential characteristic can be informally defined as a sequence that consists of plaintext differences, ciphertext differences, input-output differences of each round. It pushes the knowledge of the differences for a desired number of rounds. To attack efficiently, a high probability characteristic should be found. So, constructing a high probability one round characteristic is important. This can be achieved by observing the high probability occurances of difference pairs. Then, combining the single round characteristics in a logical way, one can obtain the sufficient number of rounds characteristic for the attack. The following is an example of a one round characteristic of the sample cipher with probability 1:

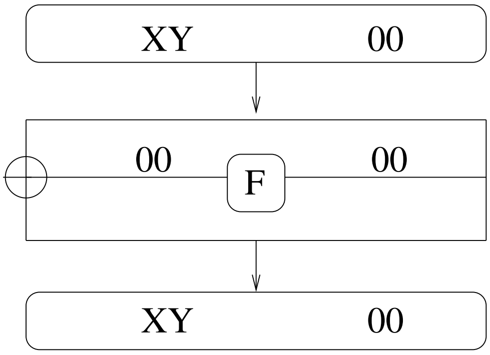


Figure 2.1: One Round Differential Characteristic for Sample Cipher

Here, X and Y are any 4 bit differences. As it is mentioned before, zero input difference of round function, say f function, consequently of S-boxes requires zero output difference. So, this one round characteristic is always true.

To find another high probability one round characteristic of the sample cipher, examine DDT (table 2.3). 8 is the greatest entry after 16. To form the characteristic in figure 2.1, the input and output differences corresponding to 16 is considered. For the following characteristic given in figure 2.2 the input-output differences corresponding to 8 on table 2.3 is used.

Observing the operations of the round function, this can be seen more clearly (figure 2.3). The

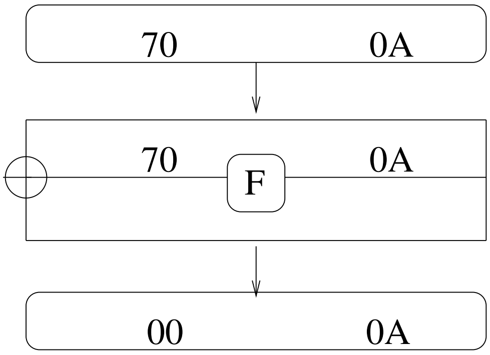


Figure 2.2: One Round Differential Characteristic for Sample Cipher

input difference to the S-boxes is 00001010 because the subkeys do not have an effect on the differences. Analysing DDT, B occurs with probability $8/16 = 1/2$ when the input difference is A. So, the output difference of S_2 is chosen as B. The nonzero bit positions of the output of the S-boxes are 5, 7 and 8. Then, the output of the f function is 00001011 because positions of nonzero bits are 2, 3 and 4 after the permutation.

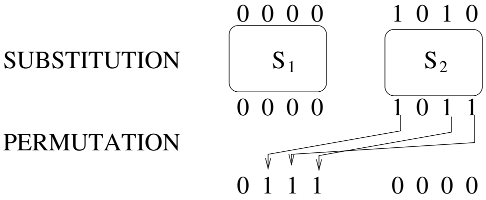


Figure 2.3: Operations of The Round Function

2.1.3 Constructing a Three Round Differential Characteristic

To find a characteristic with a certain number of rounds, the attacker seeks high numbers of occurences of output differences of S-boxes for each round, then combines the input differences and output differences of the rounds.

Once one round characteristic with a high probability is found, it is easy to construct a three round characteristic. Now, it is time to mention about a simple, general trick. Consider the one round characteristic shown in figure 2.4. Here, R is the right half of the plaintext difference and L is the output difference of the f function supposed to occur with a high probability.

This characteristic can be combined with the one shown in figure 2.1 to get a high probability three round characteristic. The input difference of the third round can be computed by XOR-

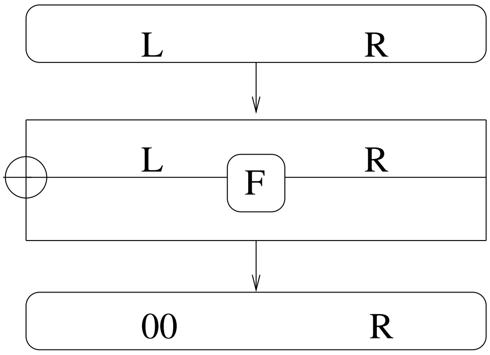


Figure 2.4: One Round Differential Characteristic

ing the right half of the plaintext, R and the output difference of the f function in the second round, which is zero. Then, choosing the output difference of the third round as L, one gets the following characteristic illustrated in figure 2.5.

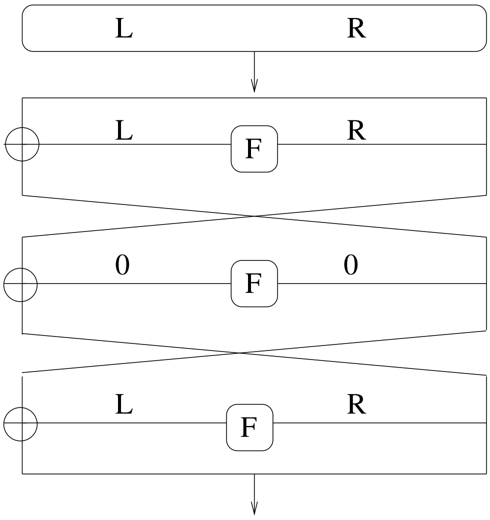


Figure 2.5: Three Round Differential Characteristic for Sample Cipher

Construction of this three round characteristic can be explained simply by adding a characteristic with probability one between the two identical high probability characteristics. This technique can be again used to create the high probability three round characteristic shown in figure 2.6 by using the one round characteristic shown in figure 2.2.

Definition 2.1.1 *Assume Λ is a cryptosystem consisting of m-bit block size. Consider an n-bit string (α_1, α_2,..., α_n) where each α_i is a tuple of m/2 bit numbers α^i_{in} and α^i_{out}. That is, α_i=(α^i_{in}, α^i_{out}). A differential characteristic is a sequence of n elements, (α_1, α_2,..., α_n), each of which*

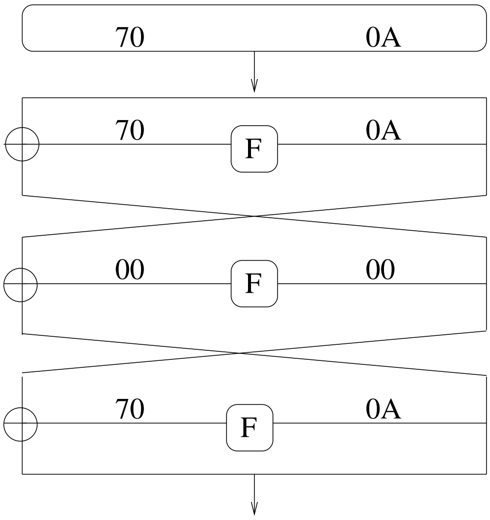


Figure 2.6: Three Round Differential Characteristic for Sample Cipher

satisfies the following properties:

- α_{in}^{1} *is the right half of the plaintext difference.*
- α_{in}^{2} *is the XOR of* α_{out}^{1} *and left half of the plaintext difference.*
- α_{in}^{n} *is the right half of the ciphertext difference.*
- α_{in}^{n-1} *is the XOR of* α_{out}^{n} *and the left half of the ciphertext difference.*
- $\alpha_{out}^{i} = \alpha_{in}^{i-1} \oplus \alpha_{in}^{i+1}$*, for all* $i \in \{2, 3, ..., n-1\}$

2.1.4 The Probability of a Differential Characteristic

Definition 2.1.2 *The S-box of a characteristic is called an active S-box, if it has a nonzero input difference.*

From the previous sections, we have seen that the probability of an active S-box is computed by using DDT. The probability of the overall characteristic, on the other hand, is computed by simply multiplying the probabilities of active S-boxes of single rounds, provided that the occurances of difference pairs in each active S-box are independent [4].

To illustrate, the computation of the probability of the three round characteristic shown in figure 2.6 will be given. In the first and the last round, S_2 is the only active S-box. Every input difference is zero in the second round. The probabilities of both the first and the third round were computed as $8/16 = 1/2$ and probability of the second round is 1. So, the overall probability for this three round characteristic is $1/2 \times 1 \times 1/2 = 1/4$. Also note that, since 8 is the unique highest entry after 16 on DDT, table 2.3, this characteristic is one of the highest probability three round differential characteristic for the sample cipher. Another characteristic having this probability can be created by choosing S_1 as the active S-boxes in the first and the third rounds.

2.1.5 Expanding the Three Round Characteristic

To break the full round sample cipher, a four round high probability differential characteristic is needed because the subkey of the fifth round will be determined. To achieve this, one more round will be added to the three round characteristic presented in figure 2.6. The output difference of f function in the fourth round can be determined by observing DDT.

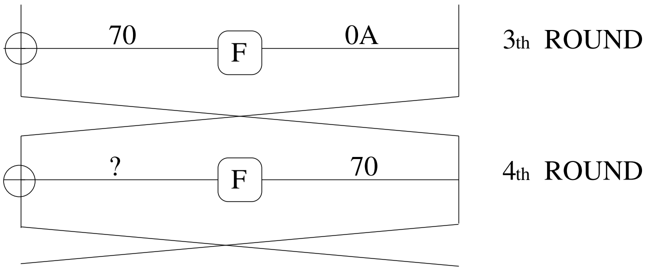


Figure 2.7: Third and Fourth Rounds of the Characteristic

Investigating DDT (table 2.3), one can see that when the particular input difference is 7, the output difference D occurs with the highest probability which is $4/16 = 1/4$. So, the output difference of the S-box in the fourth round is chosen as 11010000. The nonzero bits of the output difference of the S-box is 1, 2 and 4. After the permutation, one can see that the positions of nonzero bits become 5, 6 and 7. So, the output of the f function of the fourth round is 0E. Hence, the four round differential characteristic can be illustrated in figure 2.8.

The probability of the overall characteristic is $1/2 \cdot 1/2 \cdot 1/4 = 1/(2^4)$. Before explaining the attack, we need to conclude some remarks.

Remarks:

1. Assume the highest number on DDT is not unique. Then, starting with both of the one

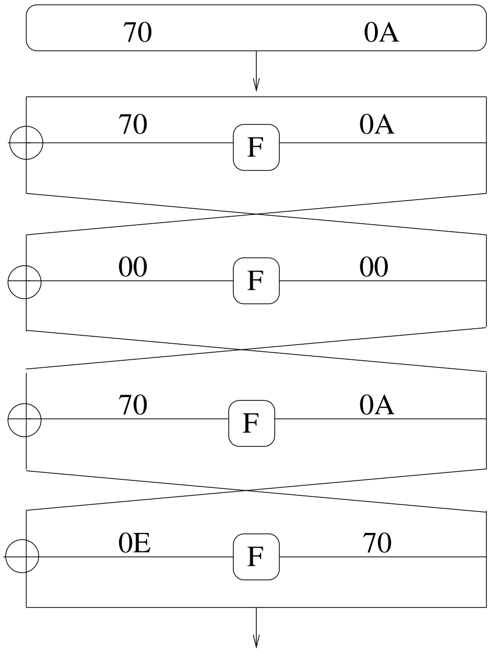


Figure 2.8: Four Round Differential Characteristic

round characteristics obtained from the highest probabilities individually, try to create characteristics with the desired number of rounds. Among these two newly constructed characteristics, choose the one that has the highest probability.

2. While finding a characteristic for the sample cipher, the greatest number appearing on DDT is chosen to start with. Nevertheless, the attacker sometimes can get a higher probability characteristic for the whole cipher by not beginning with the output difference of the S-box with the highest occurance. So, the starting point should be chosen carefully. The overall probability of the characteristic that is used for the attack should be the highest.

3. Although it is known that both of the two S-boxes are the same for the sample cipher, the second S-box in the first round is chosen as the active S-box for the characteristic. Due to the permutation component and the distribution of numbers on DDT for the sample cipher, it does not change the probability if one starts to create the characteristic by choosing the plaintext with the right half as $0A$ or $A0$. However, this is usually not the case for other ciphers. Generally, permutation components are constructed in such a way that each bit of the output of an S-box in the previous round become inputs of different S-boxes. The attacker should choose the S-box whose output difference

become an input difference of least number of S-boxes in the next round because the probability gets lower while the number of active S-boxes gets higher.

2.1.6 Differential Attack on Sample Cipher

Determining the Number of Plaintext-Ciphertext Pairs for the Attack

Definition 2.1.3 *A right pair with respect to a characteristic is a plaintext-ciphertext pair such that the intermediate differences are the same with the values specified by the characteristic. A pair which is not a right pair will be referred as a wrong pair.*

For instance, consider an n-round characteristic with plaintext difference ΔP, output difference of the n^{th} round ΔO, the i^{th} round input difference of f function ΔX_i and the i^{th} round output difference of f function ΔY_i. A plaintext-ciphertext pair is called a right pair for this n round characteristic if the plaintext difference equals to ΔP, n^{th} round output difference equals to ΔO, the i^{th} round input difference of f function is ΔX_i and i^{th} round output difference of f function is ΔY_i.

For the attack, enough number of plaintext pairs that guarantees the existence of several right pairs are needed to be collected. The number of pairs needed depends on the probability of the characteristic, the number of bits of the subkey that is desired to be determined and the level of identification of the wrong pairs among the pairs collected [2].

Consider a characteristic with probability p. For about every $1/p$ pairs an occurance of one right pair is expected, because the probability of a characteristic is the probability of the pair whose plaintext difference equals to characteristic's plaintext difference to be a right pair. Actually, note that this probability depends on the values of the keys. Namely, it changes for different values of the keys but the assumption is that the characeristic's probability is a very good approximation of it [2].

Since the right subkey encrypts significantly greater number of right pairs correctly than the wrong ones, theoretically $1/p$ pairs are enough to mount an attack but in practice, it is chosen as $c \times 1/p$ where c is a small constant. This is due to the reason that the attacker cannot determine the right pairs exactly. So, it is crucial to eliminate as many wrong pairs as possible from the chosen plaintext-ciphertext pairs for the attack. For this purpose, the right half of the ciphertext differences are checked whether they are the intended difference corresponding to the input difference of the last round. The ones, which does not satisfy the certain difference, are eliminated. This elimination ensures the release of the pairs which are obviously not a right pair. Yet, this sifting method leaves a mixture of right and wrong pairs because the attacker cannot determine the intermediate differences and so, the wrong pairs which has intermediate differences different from the ones in characeristic, cannot be eliminated. This gives rise to the following definition.

Definition 2.1.4 *The plaintext-ciphertext pair which have the differences in their plaintexts and ciphertexts obeying to the prescribed differential path but an unpredicted intermediate values by the characteristic is called a noise.*

To find a good estimate for the data requirements on the correct success rate the signal to noise ratio denoted by S/N is a great tool.

Definition 2.1.5 *The signal to noise ratio, denoted by S/N, is defined as the ratio between the right pairs and an average count in a counting scheme [2]. In other words, it is the ratio of the probability of a right pair to the probability of the noise.*

$$S/N = \frac{2^k \cdot p}{\alpha \cdot \beta}$$

where k is the number of subkey bits, p is the characteristics probability, α is the average count per counted pair and β is the fraction of the counted pairs among all the pairs.

If S/N is greatly higher than one, then only a few pairs are needed to attack. The lower the value of S/N, the higher the need of the pairs required, because the majority of the right pairs provides a better detection of the correct key value.

The Attack

Once the number of pairs is determined, the attacker collects that number of pairs whose plaintext difference is ΔP. Since differential cryptanalysis is a chosen plaintext attack, one should be able to choose as many plaintext-ciphertext pairs as needed such that the differences of pairs of the plaintexts give a particular difference. Then, compute the difference between ciphertext pairs corresponding to plaintexts pairs. By using these ciphertexts, eliminate as many wrong pairs as possible. The sifting of the wrong pairs provides a better derivation of the correct subkey value. The part of the subkey in the last round can be determined by counting the number of pairs satisfying an equation derived in the last round for each possible candidate for the subkey.

Using the four round characteristic achieved in the previous sections, a part of the subkey in the last round of the sample cipher will be obtained. The overall probability of the sample cipher is $1/(2^4)$. Choosing $c = 4$, $(1/p) \times c = 2^4 \times 2^2 = 2^6 = 64$ plaintext pairs will be enough to attack. So, collect 64 plaintext pairs whose difference gives $700A$ and get their corresponding ciphertexts. Among these, it is crucial to eliminate some of the wrong pairs. For this reason, compute the differences of the ciphertexts and check whether the right halves equal to $0A \oplus 0E = 04$. The eliminated pairs will not play a role in the rest of the attack. Let ΔC_L be the left half of the ciphertext difference and ΔC_R be the right half of the ciphertext difference.

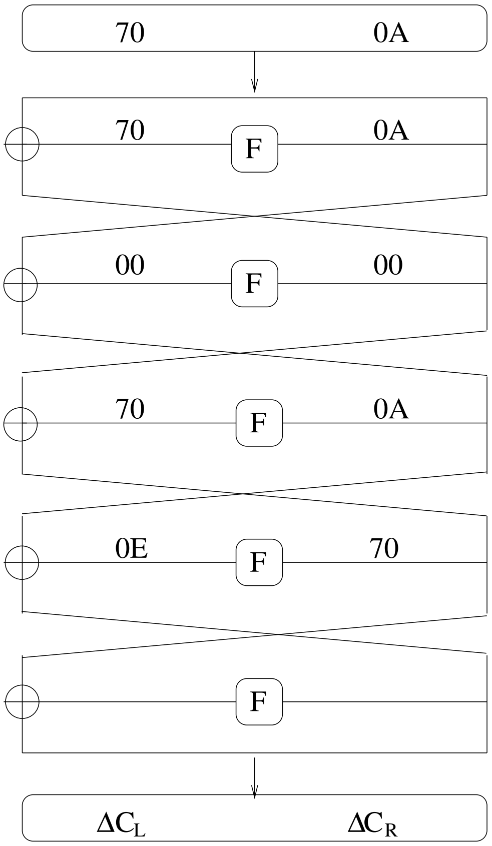


Figure 2.9: The Differential Characteristic for the Attack

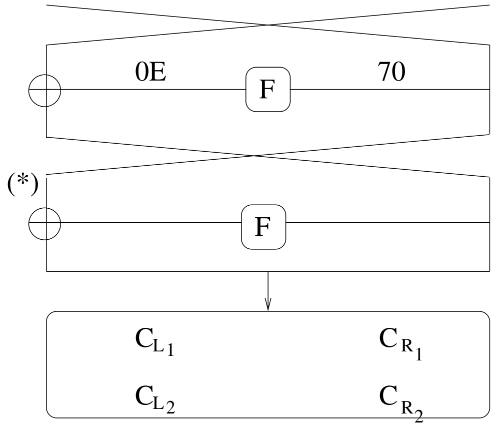


Figure 2.10: The Last Two Rounds of the Characteristic

To see the attack in more detail, focus on the last two rounds (figure 2.10). Let $\Delta C_L = C_{L_1} \oplus C_{L_2}$ and $\Delta C_R = C_{R_1} \oplus C_{R_2}$. Namely, C_{L_1}, C_{L_2}, C_{R_1} and C_{R_2} denote the left half of the first ciphertext, left half of the second ciphertext, right half of the first ciphertext, right half of the second ciphertext, respectively. Since differential cryptanalysis is a chosen plaintext attack, these values are known. The input difference of f function in the last round is ΔC_R. Also, notice that the output difference of the f function equals to

$$f(C_{R_1}, K) \oplus f(C_{R_2}, K),$$

where K is the subkey of the fifth round. By the XOR operation of the last round the equation, marked by (**) below, can be written because the difference marked with (*) in figure 2.10 is 70 and left side of the ciphertexts are also known. Notice that, the only unknown value in (**) is the subkey, K.

$70 \oplus \Delta C_L = f(C_{R_1}, K) \oplus f(C_{R_2}, K)(**)$

Input difference of the f function in the last round, namely ΔC_R is 04. One can see this by XORing the input difference of f function in the third round $0A$ and the output difference of f function of the fourth round $0E$. In the last round, input difference of the first S-box is zero. So the differential characteristic only affects the inputs to the second S-box of the last round. Hence, the last 4 bits of the subkey in the last round will be determined.

There are $2^4 = 16$ possible values of the 4 bits of the subkey in fifth round so 16 candidate subkeys $k_1, k_2, ...k_{16}$ exists. For each candidate key a count is kept. At the beginning, all counts indicate zero. Firstly, for k_1, take the first ciphertext pair and check whether the equation $(**)$ is satisfied or not. If it is satisfied, then increment the count of k_1 by one. If it is not satisfied, do not increment the count and take the second pair to do the same procedure. For k_1, check the satisfaction of the equation $(**)$ for all ciphertext pairs and keep the count of it. Determine the counts of k_i, for all i. Generally, the candidate key which has the highest count is the right key. It is not difficult to see why this is so. It is known that the differential characteristic has a high probability. Remembering that a wrong key is just a random guess, one can see that probability of the last round of the characteristic to occur with the correct key is higher than the one with the wrong key. An attacker cannot expect that the number of satisfaction of the $(**)$ with the incorrect key more than the one with the right key, since all right pairs with the correct key satisfies $(**)$. Sometimes the pairs which are wrong pair also satisfy $(**)$ by using the correct key. So, the right key satisfies $(**)$ with the probability of the characteristic plus the probability of the satisfaction of $(**)$ with the wrong pairs and the right key [4, 2]. However, sometimes wrong keys count more than the right keys. This can be due to the following reasons [4]:

- S-box properties
- The impression of the independence assumption required for determination of the characteristic probability.
- The concept of differentials are composed of multiple differential characteristics.

Remarks

1. Probability of the overall differential characteristic with n bit keys should not be lower than $1/(2^n)$ because otherwise the key can be found by an exhaustive search and there is no point to find the key using the characteristic.
2. The last 4 bits of the sample cipher is determined. To get the rest of the bits one can try to find another characteristic such that the first S-box of the fifth round is affected this time. Moreover, the attacker can do exhaustive search to find the first 4 bits.

   In addition, a known plaintext attack can be mounted using the plaintext-ciphertext pairs used in the differential cryptanalysis. For each value of the first 4 bits of the subkey, concatenate the last 4 bits of the subkey that is just determined. Give a number to each of the concatenated structure from 1 to 16 since 4 bits are unknown. Then, keep a count for each of them likewise in differential cryptanalysis. Using each k_i, $i = 1, 2, ...16$, encrypt a suitable number of plaintexts. If the encrypted data gives the corresponding ciphertexts of the plaintexts, increment the count of the corresponding k_i, where $i = 1, 2, ..., 16$. The concatenated key which has the highest count is said to be the correct key. Since the keys of all rounds are assumed to be the same, the subkey of all rounds are determined by this way.

3. Each differential characteristic provides a determination of certain bits of the key. Trying to get all bits of the key from one characteristic can be possible. This ensures less need of data and good identification of the correct key value. However, the attacker requires a huge memory to keep the counts of each key which can make the attack impractical. So, it is a good way to find parts of the subkeys using each characteristic.

2.2 Applications to Main Cryptographic Structures

Differential cryptanalysis is a very significant way of attack, that had been performed on various block ciphers, stream ciphers and cryptographic hash functions. In this chapter, differential cryptanalysis of DES and Feal-8 [2, 5, 11], which are already published before, will be studied. These are very basic attacks and useful for a learner. Also, in the next chapter, brief summaries of basic publications that includes differential cryptanalysis applied to any other block ciphers will be presented. To see the details of the briefs, it is suggested to see the references.

2.2.1 Differential Cryptanalysis of DES

Description of DES

DES was developed at IBM, as a modification of an earlier system known as Lucifer and is based on a symmetric-key algorithm that uses a 56-bit key. It has been center of interest of cryptanalysts for years. Today, it is considered to be insecure for many applications but its successors such as Triple DES is still being used.

The cryptographic algorithm transforms a 64-bit binary value into a unique 64-bit binary value based on a 56-bit variable. It is a Feistel type block cipher running 16 rounds. A block to be

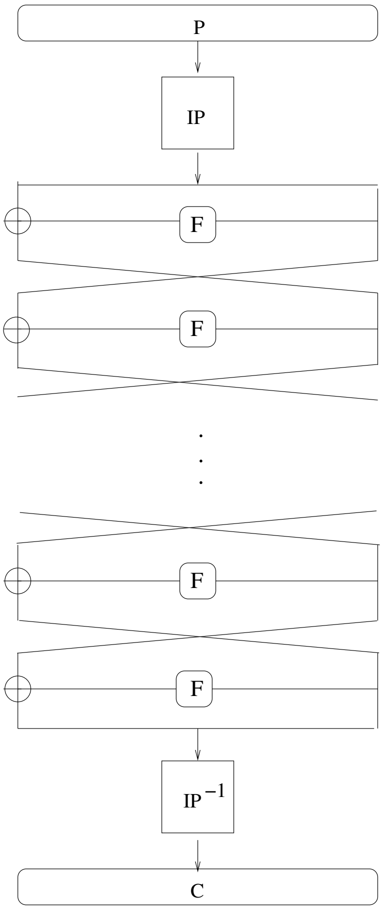


Figure 2.11: The Diagram of DES

enciphered is subjected to an initial permutation IP and then, to a complex key-dependent computation and finally to a permutation which is the inverse of the initial permutation, IP^{-1}. The round function, f, is illustrated in figure 2.12. The 32-bit data first expanded to 48-bits according to a fixed expansion function E to be able to be XORed with 48-bit subkey. Next, the 48 bit data is divided into eight parts and each of these six bit data enters a 6×4 S-box. This component is the only one to gather the nonlinearity. After the substitution, a permutation is applied on the 32 bit output of the S-boxes. Permutation is an operation performed by a fixed function, which moves an element at place j to the place k where $1 \leq j, k \leq 32$. After the permutation, 32-bit output of the round function is obtained. The exact definitions and tables for E, P and the S-boxes can be found in the original proposal [1].

Differential Attack on DES

Differential Cryptanalysis is first introduced in [2] by Eli Biham and Adi Shamir. In this paper, it is shown that any reduced variant of DES up to 15 rounds can be broken faster than

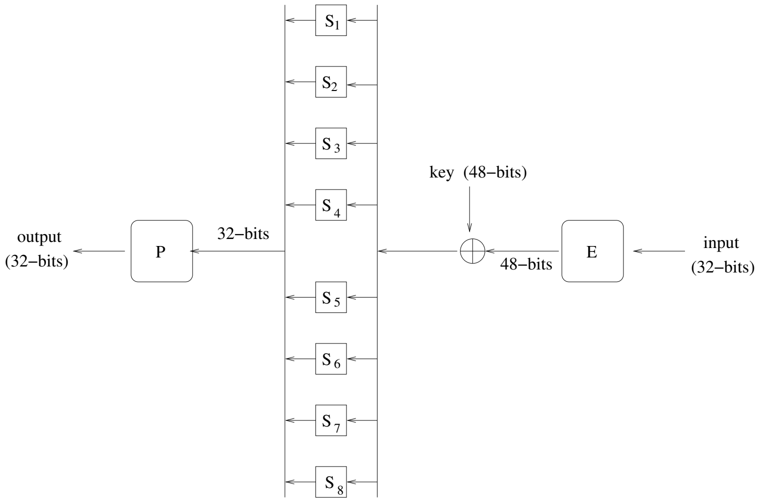


Figure 2.12: The Round Function of DES

exhaustive search and the method can be applicable to DES-like cryptosystems. Concepts of zero round (0R), one round (1R), two round (2R) and three round (3R) attacks are presented and illustrations of them are given.

Definition 2.2.1 *A* 3*R attack is such a type of attack in which the attacker can mount an n round attack using an* $n-3$ *round characteristic. The bits of the subkey that can be found in the* n^{th} *round correspond to the bits which give zero output differences in the* $(n-2)^{nd}$ *round.*

Definition 2.2.2 *A* 2*R attack is such a type of attack in which the attacker can mount an n round attack using an* $n-2$ *round characteristic.*

To mount an attack on DES reduced to four, six, eight and nine rounds a 3R-attack is applied. We will investigate all these attacks.

Differential Attack on DES Reduced to Four Rounds

To mount a four round attack, one round characteristic illusrated in figure 2.13 with probability one is used. It is aimed to determine the subkey of the fourth round.

As it is seen from figure 2.14 the output of the f function in the fourth round, marked as X is the XOR of the right half of the plaintext, left half of the ciphertext and the output of the f function in the second round. Since the input to the f function in the second round is

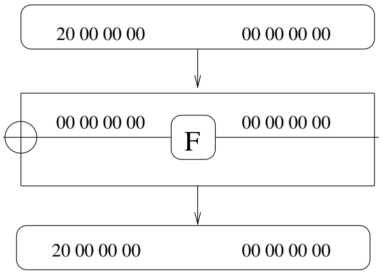


Figure 2.13: One Round Differential Characteristic for DES

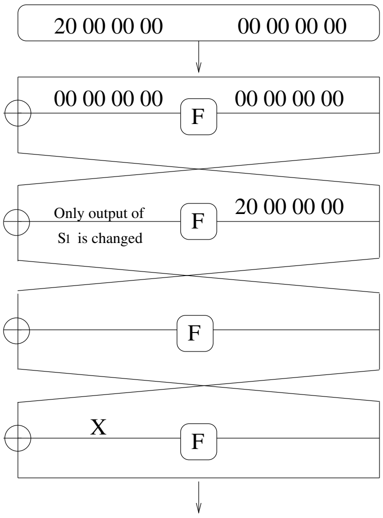


Figure 2.14: Differential Characteristic to Attack on DES Reduced to Four Rounds

20000000, the output of only the first S-box can change. Therefore, 28 bits of X is known noting that the plaintext and the ciphertext values are known. The input values of S-boxes in the fourth round are known since the right half of the ciphertexts are known. Moreover, the 28 bits, specifically the output of the seven S-boxes in the fourth round, are known due to the explanations before. For each of the seven S-boxes, 64 seperate counters are used to achieve the part of the subkey in the fourth round. The part of the candidate subkeys of each S-box is called a partial candidate subkey value. The procedure of extracting key bits from one of the S-boxes is follows:

1. For each of the partial candidate subkey value of the corresponding S-box, the input values of the S-box, marked by X_1 and X_2 in figure 2.15 are computed using the expanded values one by one.

2. The differences of X_1 and X_2 after the substitution are computed for each of the ex-

panded values to check whether they are equal to output difference of the S-box.

3. Since the probability of the characteristic is one, all the pairs used are right pairs and so, the correct partial subkey value is the one suggested by all the pairs.

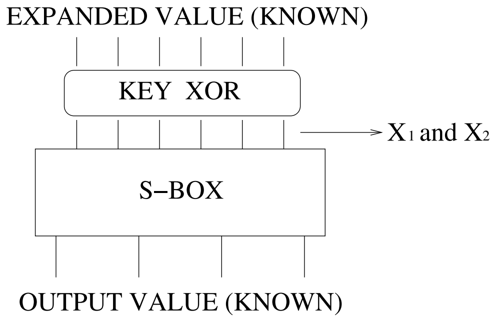


Figure 2.15: Inside the Round Function of DES

Following this procedure, 42 bits of the fourth round subkey are determined. If the subkeys are calculated via the DES key sheduling algorithm, these 42 bits are the actual key bits of DES 56 bit key and the remaining 14 bits can be found by checking all the 2^{14} possibilities of the keys whether the decrypted values of the ciphertexts via candidate keys give the particular plaintext difference or not. The correct key is the one which is suggested by all the pairs.

Differential Attack on DES Reduced to Six Rounds

To break DES reduced six rounds, two 3-round characteristics shown in figure 2.16 are used. Applying 3R attacks similar to the previous attack that is mounted on DES reduced to four rounds, 30 bits of the sixth round subkey are determined from each of these characteristics. However, the number of overall determined subkey bits of sixth round is 42 due to the common active S-boxes of the characteristics.

Differential Attack on DES Reduced to Eight and Nine Rounds

Another 3R attack is applied to break DES reduced eight rounds. In this attack, 30 bits of the subkey are determined using the five round characteristic shown in figure 2.17. The probability of this characteristic is approximately 1/55000. 18 bits of the eighth round subkey is found by a similar counting method as told previously and the remaining bits are determined by exhaustive search.

This eight round attack is extended to a nine round attack. For this, one round characteristic

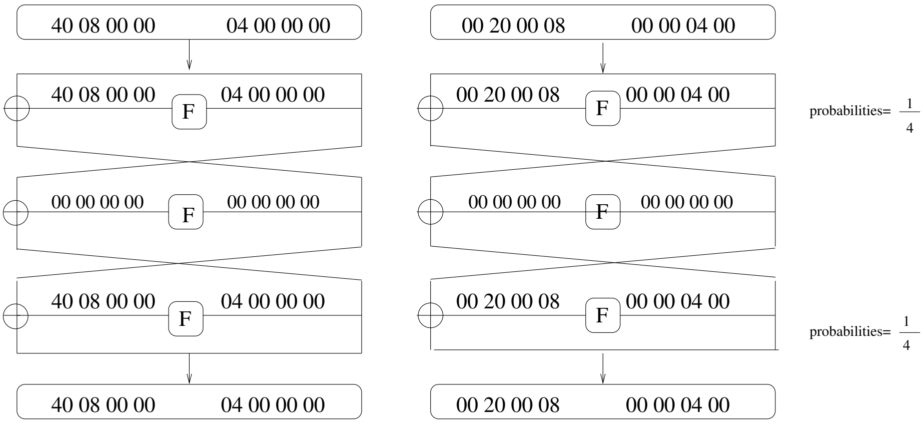


Figure 2.16: Three Round Differential Characteristics for DES

in shown in figure 2.18 is concatenated to the five round characteristic shown in figure 2.17. The total probability of this six round characterisic is approximately 1/1000000.

Moreover, using a seven round characteristic with probability 2^{-24} and applying a 2R attack another nine round attack is mounted. In this attack, more data but less memory is used than the previous nine round attack.

It is shown that 11 rounds, 13 rounds and 15 rounds DES can be broken in less than 2^{56} operations by applying 2R attacks. However, these attacks are unrealistic due to the need of great amounts of space and data.

Differential Attack on Full 16-round DES

Up to here, we have given a summary of the attacks presented in [2]. These attacks had been the most succesful attacks that can break variants of DES up to 15 rounds until 1990. Neverthless, the complexity of the attack is greater than 2^{-56} for 16-round DES. So, for breaking 16-round DES, Biham and Shamir developed their method of differential cryptanalysis [5]. In this new method, the attacker obtains possible values for the full 56-bit keys. The correct key is determined by testing the candidate keys via trial encryption, instead of using counter arrays for each possible values of the subkey bits which requires a huge memory. The crucial part of the improved attack is that, the extension of the attack from 15 rounds to 16 rounds can be achieved without decreasing the probability.

Definition 2.2.3 *A characteristic is called an iterative characteristic if the swapped value of*

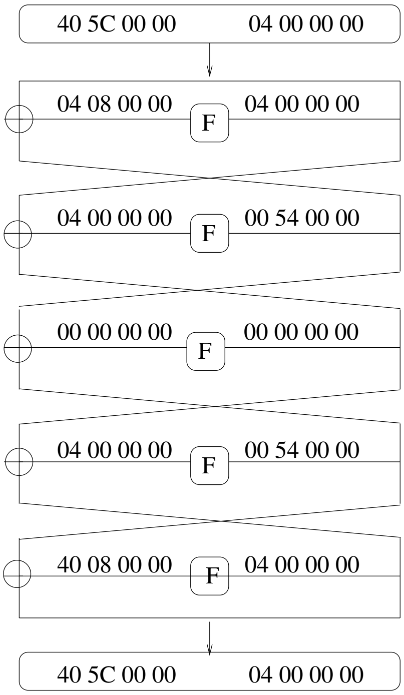


Figure 2.17: Five Round Differential Characteristic

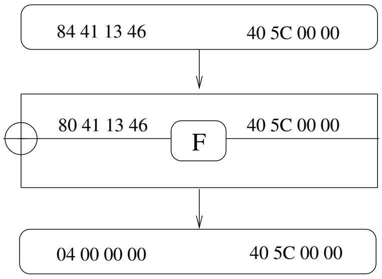


Figure 2.18: One Round Differential Characteristic

the plaintext difference equals to the ciphertext difference.

The attacker tries to build a high probability iterative characteristic such that the number of active S-boxes is minimum because the more the number of active S-boxes the less the probability. While concatenating iterative characteristics to itself, the probability decreases by a fixed rate. On the other hand, the reduction rate is usually increasing while adding rounds to a characteristic because of the avalanche effect [2].

An iterative characteristic can be easily constructed if one finds a nonzero input difference of an S-box having zero output difference with high probability. Another easy way of construction can be made when an input difference given is A, the output difference a occurs and vice versa with high probabilities. Figures 2.19 and 2.20 are the two illustrations for the constructions.

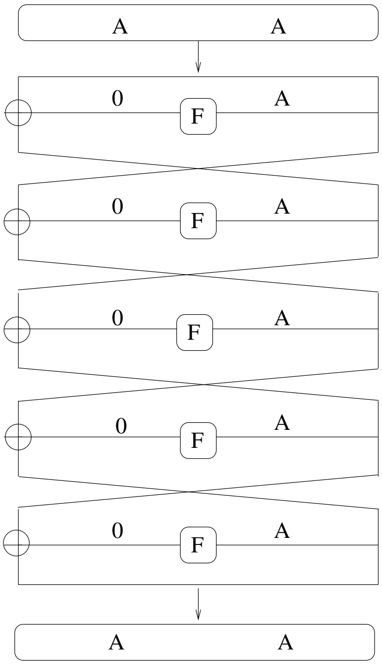


Figure 2.19: An Illustration for an Iterative Characteristic

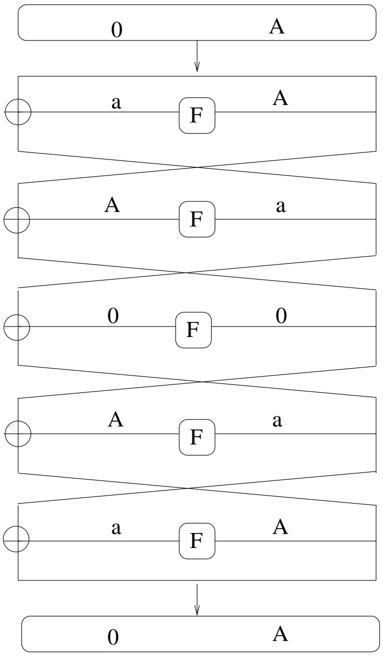


Figure 2.20: An illustration for an Iterative Characteristic

The probability of the two round characteristic shown in 2.21 is approximately 1/234. For creating a 14 round characteristic, this two round characteristic is iterated 6.5 times and then, one round is added to the top without reduction of the probability. Using this characteristic, an 16 round attack is achieved by 2R attack. The key is determined by analysing 2^{36} ciphertexts which is obtained from a pool of 2^{47} chosen plaintexts and the time complexity for the attack is 2^{37}.

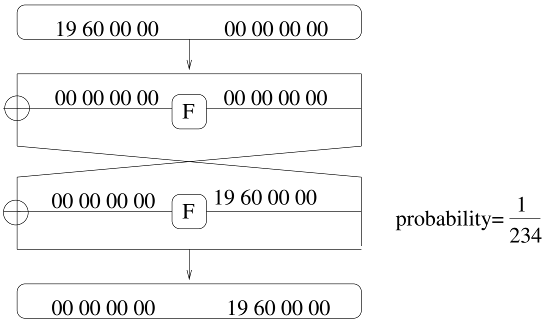


Figure 2.21: Two Round Iterative Characteristic

The probability of the characteristic of rounds from 2 to 14 is $2^{-47.2}$. The attacker tries to add one more round by defining *structures* which can be understood as an elegant chosen plaintexts in the first round such that the differences after the first round are the same with the ones in the characteristic of rounds from 2 to 14 and so that, the probability does not change. Assume, $t_1, t_2, ..., t_{4096}$ are all possible output differences of the S-boxes S_1, S_2 and S_3 leading to zero differences in remaining 20 bits and P is any arbitrary plaintext. The structure of 2^{13} plaintexts are defined as follows:

- For $1 \leq i \leq 2^{12}$, $P_i = P \oplus (t_i, 0)$ and $P_i^{'} = P_i \oplus (0, 19600000)$.
- $C_i = DES(P_i)$ and $C_i^{'} = DES(P_i^{'})$.

Using the XOR difference $(t_k, 19600000)$, one can construct 2^{24} plaintext pairs, $(P_i, P_j^{'})$ where $1 \leq i, j, k \leq 2^{12}$, as follows,

$$\begin{array}{c}
(P_1, P_1^{'}),\ (P_2, P_1^{'}),...,\ (P_{2^{12}}, P_1^{'}) \\
(P_1, P_2^{'}),\ (P_2, P_2^{'}),...,\ (P_{2^{12}}, P_2^{'}) \\
. \\
. \\
. \\
(P_1, P_{2^{12}}^{'}),\ (P_2, P_{2^{12}}^{'}),...,\ (P_{2^{12}}, P_{2^{12}}^{'})
\end{array}$$

Each of the constants, t_k, corresponds to one of the differences of the 2^{24} pairs $(P_1, P_j^{'})$, $(P_2, P_j^{'})$,...,$(P_{2^{12}}, P_j^{'})$ and so, they appear 2^{12} times in above pairs. Therefore, one can cancel the differences in the output of the round function of the first round exactly for the 2^{12} plaintext pairs which leads to offset the first round by preparing necessary plaintexts for the differential characteristic starting from the second round. Now, for each structure the probability of holding the differential characteristic is $2^{12} \times 2^{-47.12} = 2^{-35.12}$.

2.2.2 Differential Cryptanalysis of Feal-8

Description of Feal

Feal-8 is a 64 bit Feistel network block cipher running 8 rounds and presented by Akihiro

Shimizu and Shoji Miyaguchi. Although it was designed as a faster alternative to DES, it is vulnerable to various forms of cryptanalysis due to the simplicity in the round function. The diagram of Feal-8 is given in figure 2.22. As it is seen, the general view of Feal-8 is very similar to the one belonging to DES except the input and output whitening operations. At the beginning of the encryption, the data is XORed with additional subkeys, and then the left half of the data is XORed with the right half of the data. Similarly, at the end of the encryption process, the left half of the data is XORed with the right half of the data and then, the whole data is XORed with additional subkeys.

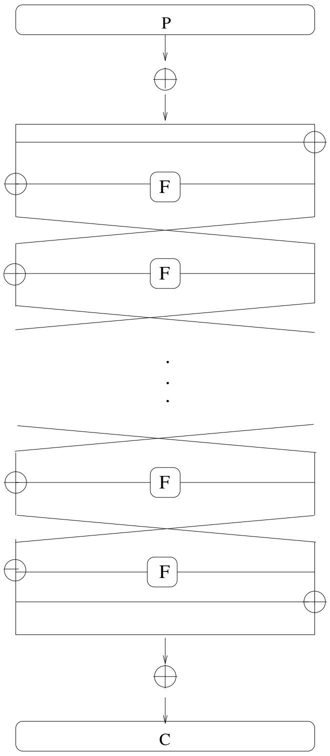


Figure 2.22: The Diagram of Feal-8

The round function of Feal-8 which is illustrated in figure 2.23 includes two different substitution boxes and the XOR operation. The S-boxes, denoted by S_0 and S_1, have 16 input bits and 8 output bits. They include rotation and addition modulo 256 operations. The following is the definition of the S-boxes:

$$S_i(x, y) := ROL2(x + y + i(mod 256)).$$

where x and y are 8-bit inputs and $ROL2(X)$ means rotation of the byte X by two bits to the left.

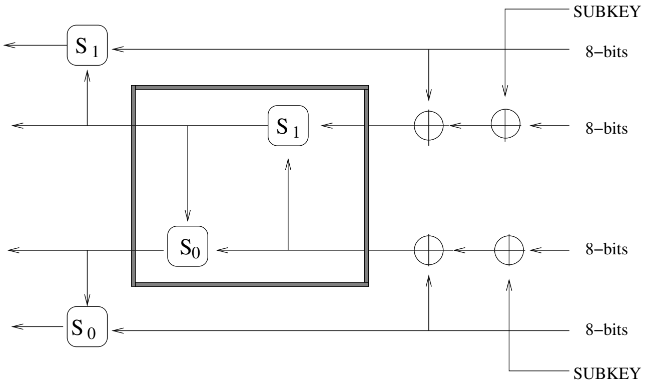


Figure 2.23: The Round Function of Feal-8

The Attack

A differential attack is presented on Feal-8 by the inventers of differential cryptanalysis in 1991 [11]. In the paper, an equivalent description of Feal-8 is given. In this description, the XOR with the subkeys in the final transformation is eliminated and the 16-bit subkeys are XORed with the middle bytes to the inputs of the round function in the various rounds are replaced by 32-bit values.

Definition 2.2.4 *[11] The actual subkeys are defined as the 32-bit subkeys of the equivalent description of Feal-8 in which the XOR with the subkeys in the final transformation is eliminated. The actual subkey which replace the subkey K_i is denoted by AK_i.*
Let $am(K)$ be the 32-bit value $(0, K_0, K_1, 0)$ where K is 16-bit long and $mx(X)$ be the 16-bit value $(X_0 \oplus X_1, X_2 \oplus X_3)$ where X is 32-bit long. Then, $mx(AK_i) = (AK_{i0} \oplus AK_{i1}, AK_{i2} \oplus AK_{i3})$ are called 16-bit actual subkeys. The actual subkey of the last round is called the last actual key.

- *The actual subkeys in the even rounds $i + 1$ are $AK_i = Kcd \oplus Kef \oplus am(K_i)$.*
- *The actual subkeys in the odd rounds $i + 1$ are $AK_i = Kcd \oplus am(K_i)$.*
- *The actual subkeys of the initial transformations are $AK89 = K89 \oplus Kcd \oplus Kef$ and $AKab = Kab \oplus Kef$.*

In this attack [11], actual subkeys are obtained instead of the subkeys themselves because XORs of the ciphertexts and internal values in the F function are found.
The byte addition operation of the S-box component is not XOR linear. This can be shown

by a simple example:

Example: Let $f(x, y) = x + y(mod16)$. At least for one (x, y) and (z, k), it is needed to show that $f((x, y) \oplus (z, k)) \neq f(x, y) \oplus f(z, k)$. Now, let $x = 1010$, $y = 0010$, $z = 0001$ and $k = 0011$ which are not written in hexadecimal form. Then, $f((x, y) \oplus (z, k)) = f(x \oplus z, y \oplus k) = f(1011, 0001) = 1100$. On the other hand, $f(x, y) \oplus f(z, k) = 1100 \oplus 0100 = 1000$. This can be easily extended to a byte addition mod 256. So, we can conclude that the byte addition operation is not XOR linear since $1100 \neq 1000$.

Also, the byte addition operation is the only non-linear operation in Feal-8 and so, the security of Feal-8 depends on the security of this operation. However, there are very simple weaknesses in this operation. This can be easily investigated by observing the S-boxes.

It is difficult to observe the XOR tables of the S-boxes directly, since they are so large, namely 16×8. Instead, the probabilities for combination of the two S-boxes shown with a rectangle in figure 2.23 are analysed and some properties of S-boxes are investigated. To illustrate, if the input difference of the combination shown with a rectangle in figure 2.23 is 0000, the output difference is always 0000. Similarly, if the input difference is 8080, the output difference becomes 0002 with probability 1. This can be more understandable by the following example.

Example:

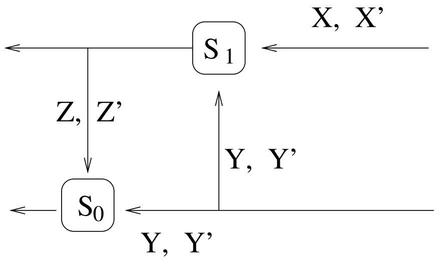


Figure 2.24: Combination of the Two S-Boxes

In figure 2.24, combination of the two S-boxes shown with a rectangle in figure 2.23 is illustrated. X, Y, X' and Y' be inputs of the S-boxes. Let $X = 81$, $X' = 01$, $Y = 82$ and $Y' = 02$. Notice that the concatenation of $X \oplus X'$ and $Y \oplus Y'$ equals to 8080. Now, we will check whether the output difference is 0000 or not.

- For $X = 81$ and $Y = 82$, we have $S_1(10000001, 10000010) = 10000001 + 10000010 + 00000001 = 00010000 = Z$ which equals to 10 in hexadecimal notation.

- For $X' = 01$ and $Y' = 02$, we have $S_1(00000001, 00000010) = 00000001 + 00000010 +$

00000001 = 00010000 = Z' which equals to 10 in hexadecimal notation.

So, the outputs of S_1 are $Z = Z' = 10$ whose XOR is 00. Also, the inputs of S_0 are $Z = 10$, $Z' = 10$, $Y = 82$ and $Y' = 02$. Similar to the above computations we compute the output difference of S_0.

- For $Z = 10$ and $Y = 82$, we have $S_0(00010000, 10000010) = 00010000 \oplus 10000010 = 01001010$ which equals to $4A$.
- For $Z' = 10$ and $Y' = 02$ we have $S_0(00010000, 00000010) = 00010000 \oplus 00000010 = 01001000$ which equals to 48.

The output difference of S_0 is $4A \oplus 48 = 02$. So, by concatenating the output difference of S_1 which is 00 and S_0 which is 02, one finds the output of the combination of S-boxes shown in figure 2.24 is 0002.
Moreover, the following properties are always satisfied for both of the S-boxes letting Δx and Δy be input differences, Δz be the output difference and S is the substitution function, namely S_0 or S_1:

- $S(\Delta x = 80, \Delta y = 80) = \Delta z = 00$.
- $S(\Delta x = 80, \Delta y = 00) = \Delta z = 02$.

The characteristics are constructed using the observations that are explained up to here. There exists non-trivial one round characteristics. Using one of these, an illustration of a three round characteristic with probability one is given in figure 2.25. There exists two more non-trivial three round characteristic with probability 1.

This characteristic is extended to six rounds by adding one extra round between the first and the second round, between the second and the last round and at the end of the last round as it can be seen from figure 2.26. The probability of this characteristic is $1/128$ and the left half of the output data is not fixed to an exact value.

In figure 2.26, $X \in \{5, 6, 7, 9, A, B, D, E, F\}$, $Y \in \{9, A, B\}$, $Z \in \{0, 1, 3\}$ and $W = X \oplus 8$. To break the full rounds approximately 1000 pairs of ciphertexts whose corresponding plaintexts differences equal to $A200800022808000$ are used. The plaintexts are motivated by the six

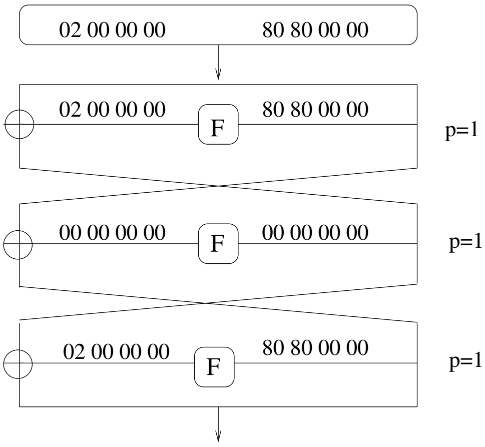


Figure 2.25: Three Round Differential Characteristic for Feal-8

round characteristic with probability 1/128, shown in figure 2.26. Calculating the exact value of the subkeys from the actual keys is also shown in the paper.

2.3 Generalizations of Differential Cryptanalysis

After the foundation of differential cryptanalysis, the cryptographers design ciphers resistant to this type of attack. Hence, some generalizations are made to develop the cryptanalysis method. Namely, the higher order differential cryptanalysis and truncated differential cryptanalysis are introduced. In this section, these two methods of cryptanalysis will be explained and brief applications of them will be given.

2.3.1 Higher Order Differential Cryptanalysis

Lai introduced the higher order derivatives of multi-variable functions in 1994 [14]. Although the attack is a generalization of the ordinary differential cryptanalysis, it can be seen as a type of algebraic attack. In the basic concept of differential cryptanalysis, one considers the difference of only two plaintexts, whereas the difference of more than two plaintexts are also mentioned in the notion of higher order differential cryptanalysis. The attacks based on this notion is applicable to the ciphers that can be expressible by Boolean polynomials with low degree.

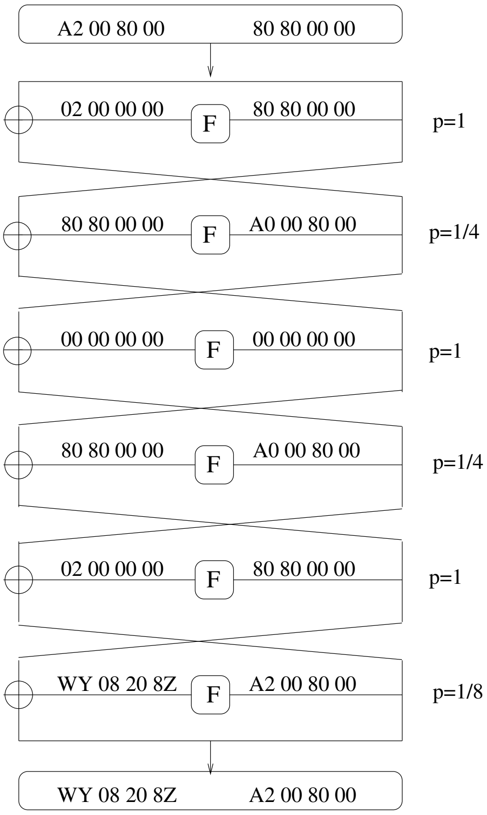


Figure 2.26: Six Round Differential Characteristic for Feal-8

Definition 2.3.1 *[14] Let $(S,+)$ and $(T,+)$ be Abelian groups. For a function $f{:}S \rightarrow T$, the derivative of f at the point $a \in S$ is defined as*

$$\Delta_a f(x){=}f(x+a) - f(x).$$

The i^{th} derivative of f at the point $a_1, a_2, ...a_i$ is defined as

$$\Delta^{(i)}_{a_1,...,a_i} f(x){=}\Delta_{a_i}(\Delta^{(i-1)}_{a_1,...,a_{i-1}} f(x)).$$

Proposition 2.3.2 *[14] Let $L[a_1, a_2, ..., a_i]$ be the list of all 2^i possible linear combinations of a_1,a_2,...,a_i. Then,*

$$\Delta^{(i)}_{a_1,...,a_i} f(x) = \sum_{\gamma \in L(a_1,a_2,...,a_i)} f(x+\gamma)$$

If a_i is linearly dependent of a_1,a_2,...,a_{i-1}, then

$$\Delta^{(i)}_{a_1,...,a_i} f(x) = 0.$$

Proposition 2.3.3 *[14] Let deg(f) denote the nonlinear degree of a multivariable polynomial function $f(x)$. Then,*

$$deg(\Delta_a f(x)) \leq deg(f(x)) - 1$$

The first attack based on the notion of higher order derivatives is proposed by Knudsen in [16]. It is applied to a sample cipher whose round function does not leak any partial information for any nontrivial difference. This 5 round cipher, which is secure against differential cryptanalysis, can be broken by higher order differential cryptanalysis.
Then, an attack on KN cipher is proposed by Knudsen and Jakobsen [17]. This attack exploits the low degree of the round function of the KN cipher. In the following section, this attack is summarized.

2.3.1.1 Higher Order Differential Cryptanalysis of the KN Cipher

In this section, a brief description of the KN cipher is given and an attack on this cipher, proposed in [17] is summarized.

1. **The KN Cipher**

   The KN cipher is proposed by Nyberg and Knudsen [23]. It is designed to be provably secure against ordinary differential cryptanalysis. In [24], it has proven that the cipher is also secure against linear cryptanalysis. However, the cipher can be broken by using higher order differentials due to its low degree round function.
   The cipher is a Feistel Network structure with 64 bit block-size. It is suggested to be used with at least 6 rounds. Here, some definitions and notations are given to define the round function:

   - $k^i = \left\{k^i_1, k^i_2, ..., k^i_{33}\right\} \in GF(2^{33})$ is the i^{th} round subkey.
   - $x^i = \left\{x^i_1, x^i_2, ..., x^i_{32}\right\} \in GF(2^{32})$ is the i^{th} round input.
   - $e : GF(2^{32}) \rightarrow GF(2^{33})$ is the function extending the argument by concatenation with an affine combination of the input bits.
   - $f : GF(2^{33}) \rightarrow GF(2^{33})$ is a cubing function defined as $f(x) = x^3$.
   - $d : GF(2^{33}) \rightarrow GF(2^{32})$ is a function which discards one bit from its argument.

The round function $F : GF(2^{32}) \rightarrow GF(2^{32})$ is defined as follows (see figure 2.27):

$$F(x^i, k^i) = d(f(e(x^i) \oplus k^i)).$$

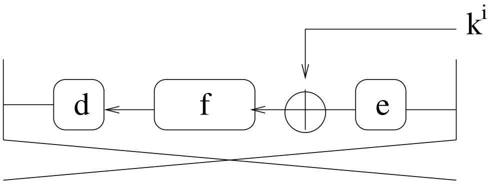


Figure 2.27: The Round Function of the KN Cipher

Notice that the output of the each round, y^i, can be seen as $y^i = F(x^i, k^i) = (h_0(x^i, k^i), h_1(x^i, k^i), ..., h_{31}(x^i, k^i))$ because F is a vector of 32 tuple of boolean functions, namely $F = (h_1, h_2, ..., h_{32})$. For each i, it is shown that $deg(h_i) = 2$ [23].

2. **The 6-Round Attack on KN Cipher**

   Although the KN cipher is provably secure against differential and linear cryptanalysis, Jacobsen and Knudsen showed that it can be broken by the higher order differential attack [17]. They exploited the low degree round function of the cipher.

   For the attack, suppose the right half of the plaintexts are kept constant, namely the zero vector in $GF(2^{32})$. Let the left hand side of the plaintext be $x = \{x_1, x_2, ..., x_{32}\}$. Since the right hand side of the plaintexts are fixed, the right hand side of the output of each round can be expressible by the subkeys of the previous rounds and x. Denote the right hand side of the output of the i^{th} round by $y_R^i[k](x)$ where k stands for the subkeys k^1, k^2,..., k^{i-1}. In addition, let $C_L(x)$ and $C_R(x)$ be boolean functions of the left half of the input x, corresponding to the left and right hand halves of ciphertext, respectively. The attack is mounted in the light of the following proposition.

Proposition 2.3.4 *Let $\{a_1, a_2, ..., a_{d+1}\}$ be a subset of $GF(2^n)$. Assume the nonlinear degree of the multivariable function, namely $f(x)$, is d. Then, $\Delta^{(d)}_{(a_1,...a_d)} f(x)$ is a constant and $\Delta^{(d+1)}_{(a_1,a_2,...,a_{d+1})} f(x)$ is zero.*

The degree of $y_R^5[k](x)$ is at most eight, since the degree of the round function is two. So, by the proposition we can say that for any subkey and any subset $\{a_1, a_2, ..., a_9\}$ of $GF(2^{32})$, $\Delta^{(9)}_{(a_1,a_2,...,a_9)} y_R^5[k](x) = 0$. Also, note that,

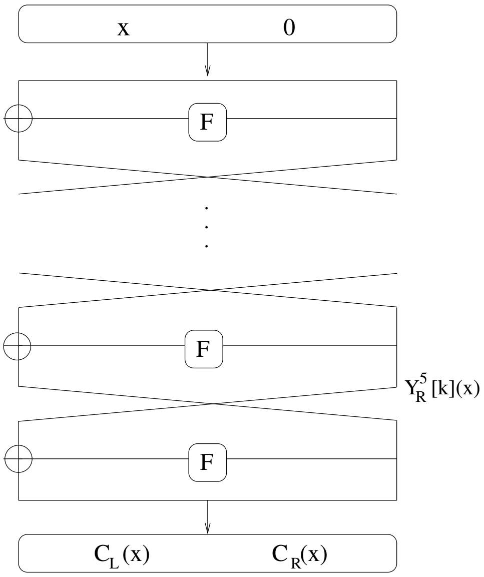


Figure 2.28: The KN cipher with 6-rounds

$$\Delta^{(9)}_{(a_1,a_2,\ldots,a_9)} y_R^5[k](x) = \sum_{x\in L(a_1,a_2,\ldots,a_9)} y_R^5[0](x)$$

.

Then, by the help of figure 2.28 and the previous discussions, one can see that,

$$\sum_{x\in L(a_1,a_2,\ldots,a_9)} y_5[0](x) = \sum_{x\in L(a_1,a_2,\ldots,a_9)} F[k^6](C_R(x)) + \sum_{x\in L(a_1,a_2,\ldots,a_9)} C_L(x) = 0.(*)$$

Similar to the ordinary differential cryptanalysis, for every possible value of the subkey, k^6, the equation marked with (*) is checked whether it holds or not. The correct key is the one that satisfies the equation. The number of required chosen plaintexts for the attack is 2^9 and the running time is 2^{41}.

2.3.2 Truncated Differential Cryptanalysis

Truncated differential cryptanalysis is first introduced by L. Knudsen [26]. Truncated differentials are roughly defined as the differentials that are partially predicted. By using this notion, one can reduce the complexity that is computed for the ordinary differential cryptanalysis because only a part of the differences are predicted.

Definition 2.3.5 *Let (P, C) be an n-round differential. (P', C') is called an n-round truncated differential, if P' is a subsequence of P and C' is a subsequence of C [26].*

In [26], an attack on DES reduced to six rounds using truncated differentials is presented. Since DES works on bits, the attack is mounted using the wide definition of truncated differentials above. However, different from the attack presented in [26], ”bytewise” truncated differentials, where one byte of the difference is regarded as 1 (non-zero) or 0 (zero), are useful in attacking byte-oriented block ciphers. To illustrate, in [25] truncated differentials are regarded subsets of the characteristics that can be used to attack the block cipher, SAFER, by using information on whether several bits of the difference are zero or not [27]. Next, we will give a brief summary of the two attacks, namely the ones presented in [26, 25].

2.3.2.1 Truncated Differential Cryptanalysis of DES reduced to six rounds

L. Knudsen presented a truncated differential attack on DES reduced to six rounds with ignored initial and final permutations using approximately 46 chosen plaintexts and 3500 encryptions [26]. The attack somehow looks similar to the Differential-Linear attack that is described in [28] in some manners but they are of course different. In figure 2.29, the 4-round differential used for the attack is presented. We denote the output of the fourth round $O = (O_L, O_R)$ where O_L and O_R stand for the left and right half of O, respectively. S_i denotes the i^{th} S-box where $1 \leq i \leq 8$.

The plaintext difference is chosen as $(\alpha, 20000000)$ and it is assumed the output of the first round is α, so that the input of the third round is 20000000. However, it is not always possible. To satisfy this with enough right pairs two sets of four plaintexts are constructed with the desired difference. Moreover, to get the exact right key, a similar differential where all quantities 20000000 are replaced by 40000000 is used and the similar construction of the sets of the plaintexts are made for this differential. For details of these constructions, please refer to [26].

As it is seen from figure 2.29, the input difference of the third round only affects S_1. So, the output of S_1 is nonzero. According to the permutation function of DES the input differences of only S_1 and S_7 are zero in the fourth round. Hence, we know the eight bits of O_L which gives us the knowledge of eight bits of the output of the last round because the ciphertexts are

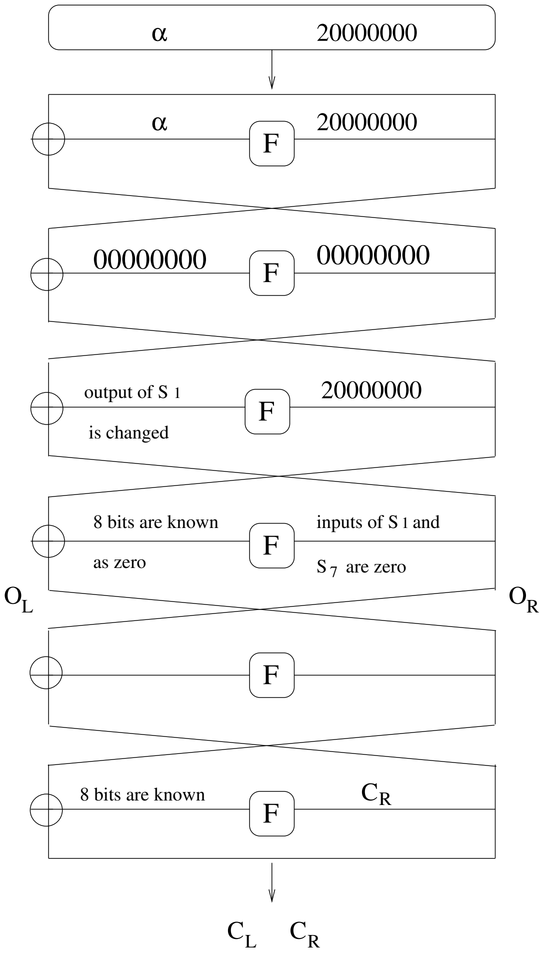


Figure 2.29: 4-round Differential of DES

also known.

The attack finds 18 key bits in total, namely 6 bits of the first round key (explained in 1) and 12 bits of the last round key (explained in 2). Let k^i_j denote the subkey bits entering the j^{th} S-box in the i^{th} round. The determination procedure of the key bits is as follows.

1. As it is mentioned before, we cannot always say that the output of the first round is α. So, by using the constructed plaintexts, part of the first round subkey is guessed. Since only the first S-box has the nonzero input difference in the first round, 6 bits of the subkey corresponding to this S-box are determined.

   - For every value of k^1_1, construct a 48 bit candidate subkey for the first round, call k^1_{cand}, by concatenating k^1_1 and 42 randomly chosen bits.
   - For each k^1_{cand}, compute the output difference of the first round using the plaintext

pairs one by one. That is, for each plaintext pair, to illustrate say for P_1 and P_2, evaluate $\Omega = f(k^1_{cand}, P_1) \oplus f(k^1_{cand}, P_2)$.

- Check whether Ω equals to α or not. If it equals, then the pair of plaintexts P_1 and P_2 is a right pair with respect to the characteristic shown in figure 2.29.
- Repeat the procedure until four right pairs are gathered.

2. The four right pairs determined in (1) will be used to obtain 12 bits of the sixth round subkey. Since we know eight bits of O_L, we can find the eight bits of the output difference of the round function in the sixth round. These bits corresponds to the outputs of the first and the seventh S-boxes. The attack is firstly mounted on the first S-box. Then, if one key value for k^6_1 is suggested by all pairs, do the attack on the seventh S-box. The attack is as follows.

   - Determine ciphertexts of all four pairs.
   - For each of the candidate subkey, determine four bit output value of the f function in sixth round by using C_R.
   - Check if this value equals to the four bit value corresponding to $C_L \oplus O_L$

2.3.2.2 Truncated Differential Cryptanalysis of SAFER

SAFER is a 64-bit, substitution-permutation network block cipher working on bytes. The cipher consists of three main layers. We denote the layer that the key is mixed with the data as key mixing, the layer where the functions X and L are applied as XL and the layer where the Pseudo-Hadamard Transformation is applied as PHT. After the last round an output transformation is applied to the outputs of the last round and then, the ciphertext is obtained. It is proven that SAFER is secure against the ordinary differential cryptanalysis [29] and linear cryptanalysis [30]. However, in [25] it is presented that SAFER reduced to five rounds is broken by using truncated differentials. For the attack, a difference of two bytes, say a and b, is defined as $a - b \pmod{256}$. Figure 2.30 shows the 5-round truncated differential with probability approximately 2^{70}. The input and output differences of all five rounds rounds and for the last two rounds the input differences of the PHT are also shown in figure 2.30.

Here N and v denote the nonzero bytes, x denotes an odd nonzero byte and z_i's denote the values which cannot be predicted. Since the output transform consist of bytewise XORing and

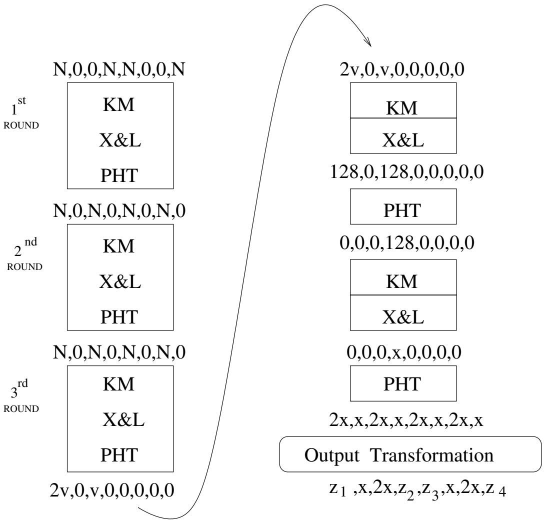


Figure 2.30: The 5-round Truncated Differential for SAFER

adding modulo 256 of the last round key, the ciphertexts of the right pairs for this differential are of the form $[z_1, x, 2x, z_2, z_3, x, 2x, z_4]$. It is not necessary to determine the value of the nonzero bytes for the attack. The important thing is to determine which bytes are zero. 2^{70} pairs are needed to get one right pair. Therefore, approximately 128 structures, a total of 2^{39} plaintexts are required. Two filtering processes that are applied on the pairs to discard the wrong ones are described.

- In the first process, the second byte of the ciphertext differences are checked if they are odd or not because it is known that x is odd. If this is so, then the differences in the third, sixth and seventh bytes are checked whether they have the desired relation between them. Notice that, for each right pair the differences in bytes 2, 3, 6 and 7 are x, $2x$, x, and $2x$ respectively. By this filtering, we are left with the 2^{45} pairs.
- For the other filtering process, the following lemma is used.

  Lemma 2.3.6 *Let x and y be two bytes and let k be a key byte. Let $z = x - y$ mod 256 and $z' = (x \oplus k) - (y \oplus k)$ mod 256. Then, the least significant bit of z and z' are equal [25].*

  According to the lemma, one can see that z_1 and z_3 must be even. Also, z_2 and z_4 must be odd since x is odd. So, check whether the z_i's have the right parity for all 2^{45} pairs. After this process, 2^{41} pairs are remained.

More filtering processes are also suggested. For details please refer to [25]. Now, each of the non-discarded pairs suggests two values for each of the first, fourth, fifth and eighth bytes of the last round subkey, on average. This gives 16 different values for the 32-bit subkey. In the attack, they used the fact that the round key byte i, $i \in \{1, 2, ..., 8\}$, in each round derived from the same key byte. Using the 16 key values, check if the plaintext differences yield a correct output difference after the first round. Every pair will suggest $16 \times 2^{-15} = 2^{-11}$ values on average of the first, fourth, fifth and eighth key bytes because after the first round, there are two possible sets of four bytes with nonzero values. In total, 2^{41} pairs suggest 2^{30} values of the four bytes of the key and so, an exhaustive search can be done in time about $1/2 \times 2^{30} \times 2^{32} = 2^{61}$. The complexity of the attack can be decreased by repeating it. To illustrate, by repeating the attack 64 times, and using 2^{45} plaintexts, the right key value is one of the $2^{32} \times 2^{-19} = 2^{13}$ most suggested values with a high probability. The complexities are given in table 2.4.

Table 2.4: Complexities of the Differential Attack on SAFER with 5 rounds

Rounds	*Time*	*Plaintexts*	*Storage*
5	2^{61}	2^{39}	2^{32}
5	2^{46}	2^{45}	2^{32}
5	2^{35}	2^{46}	2^{32}

CHAPTER 3

FURTHER ILLUSTRATIONS FOR DIFFERENTIAL CRYPTANALYSIS

Basic differential cryptanalysis and the two significant applications, namely the differential attack on DES and Feal-8 have been explained previously. In this chapter, further illustrations for differential cryptanalysis will be given. These illustrations are just summaries of the attacks. It is aimed to give some idea about the attacks and references for the reader.

3.1 Differential Cryptanalysis of Khafre

Khafre is a Feistel network, 64-bit DES-like block cipher proposed as software-oriented alternatives to DES by Ralph Merkle in 1989. It encrypts a small amount of data rapidly. It uses 8×32, key independent standard set of S-boxes and number of rounds varies among 16, 24 and 32. Key material is XORed with the 64 bit data block before the first round and thereafter every 8 rounds.

In [6], it is shown that using differential cryptanalysis, Khafre up to 24 rounds can be broken. They take the advantage of some weaknesses in the design of S-boxes. First of all, given an output difference of an S-box, it is easy to find the input pair because the input pair is usually unique. This is due to the reason that, the S-boxes are 8×32. That is the number of output bits of an S-box is more than twice the number of input bits. Observe that, the number of all possible input pairs for each S-box is $\frac{2^{8 \cdot 2}}{2} = 2^{15}$. Hence only about a small fraction, namely 2^{17} of the 32-bit values are outputs of some pair.

Another weakness is the existence of characteristics which has only one even or one odd

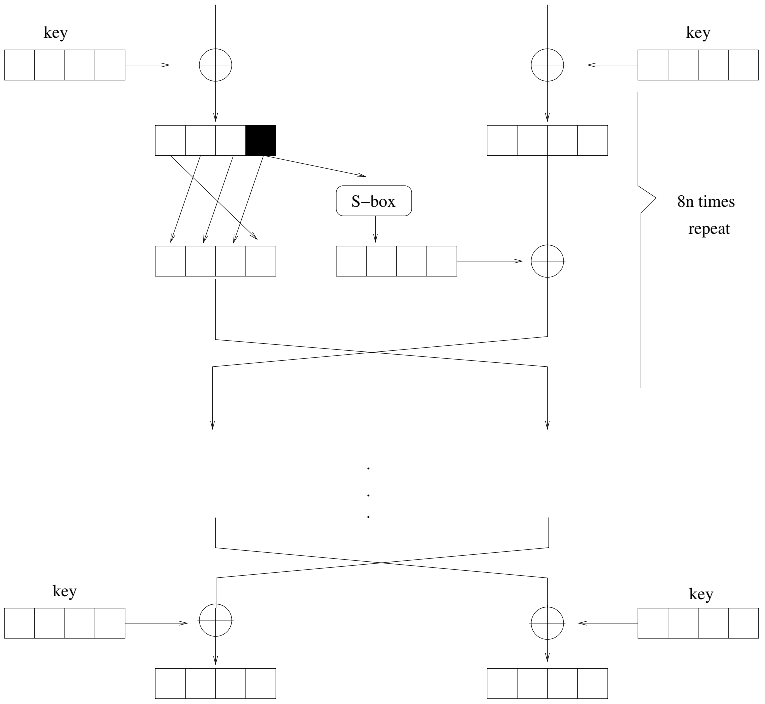


Figure 3.1: General View of Khafre

round that has nonzero input difference to the S-box. If the given pair is a right pair the output difference of this round can be easily derivable using the plaintext and the ciphertext difference.

The best characteristic for Khafre in the paper is the 16-round characteristic shown in table 3.1 with probability 2^{-16}. The number of pairs, chosen plaintexts and known plaintexts needed for the attack are 3×2^{16}, 1536 and $2^{37.5}$, respectively. Totally, 1536 encryptions are needed for chosen plaintext differential attack, while $2^{37.5}$ encryptions are needed to mount a known plaintext differential cryptanalytic attack.
For breaking the 24 rounds Khafre, a characteristic with probability 2^{-56} is used. This time 2^{60} pairs, 2^{53} chosen plaintexts and $2^{58.5}$ known plaintexts are needed.

3.2 Differential Cryptanalysis of LOKI

LOKI is a 64 bit DES-like cryptosystem first published in 1990 by Lawrie Brown, Josef Pieprzyk, and Jennifer Seberry. It uses 64 bit key and runs 16-rounds. Its general structure is similar to DES but of course it uses different versions of substitution, permutation and

Table 3.1: Characteristic for Khafre

Round	*Left Half*	*Right Half*		*Output XOR*
Ω_p	$0\,0\,A\,0$	$0\,0\,0\,0$		
1	$0\,0\,A\,0$	$0\,0\,0\,0$	→	$0\,0\,0\,0$
2	$0\,0\,0\,0$	$0\,0\,A\,0$	→	$0\,0\,0\,0$
3	$A\,0\,0\,0$	$0\,0\,0\,0$	→	$0\,0\,0\,0$
4	$0\,0\,0\,0$	$A\,0\,0\,0$	→	$0\,0\,0\,0$
5	$0\,A\,0\,0$	$0\,0\,0\,0$	→	$0\,0\,0\,0$
6	$0\,0\,0\,0$	$0\,A\,0\,0$	→	$0\,0\,0\,0$
7	$0\,0\,0\,A$	$0\,0\,0\,0$	→	$0\,0\,0\,0$
8	$0\,0\,0\,0$	$0\,0\,0\,A$	→	$B\,C\,D\,E$
9	$0\,0\,A\,0$	$B\,C\,D\,E$	→	$F\,G\,H^0 \oplus A\,I$
10	$D\,E\,B\,C$	$F\,G\,H^0\,I$	→	$J^0 \oplus D\,K^0 \oplus E\,L \oplus B'\,M \oplus C'$
11	$H^0\,I\,F\,G$	$J^0\,K^0\,L\,M$	→	$N^0 \oplus H\,P^0 \oplus I\,Q \oplus F'\,R \oplus G'$
12	$M\,J^0\,K^0\,L$	$N^0\,P^0\,Q\,R$	→	$S^0 \oplus M\,T \oplus J^{0'}\,U^0 \oplus K^0\,L'$
13	$R\,N^0\,P^0\,Q$	$S^0\,T\,U^0\,0$	→	$0\,0\,0\,0$
14	$U^0\,0\,S^0\,T$	$R\,N^0\,P^0\,Q$	→	$V^0 \oplus U^0\,W\,X^0 \oplus S^0\,T'$
15	$P^0\,Q\,R\,N^0$	$V^0\,W\,X^0\,0$	→	$0\,0\,0\,0$
16	$W\,X^0\,0\,V^0$	$P^0\,Q\,R\,N^0$	→	$Y^0 \oplus W\,Z^0 \oplus X^0\,\alpha^0\,\beta^0 \oplus V^0$
Ω_T	$Q\,R\,N^0\,P^0$	$Y^0\,Z^0\,\alpha^0\,\beta^0$		

expansion boxes.

In [6], a differential attack is mounted on LOKI. The S-boxes of LOKI are 12 × 8, so the XOR table is larger than the one belonging to S-box of DES but the values on the table are smaller. It is analysed that the maximum probability obtained from the XOR table is $1/64$ and the average probability is $1/256$. The advantage of finding an iterative characteristic is mentioned before. One can find a one round characteristic with two S-boxes having non-zero input differences resulting with zero output differences. In DES, the attacker requires at least three S-boxes to have such a characteristic. The two round characteristic shown in figure 3.2 has probability approximately $2^{-13.12}$. Iterating this characteristic to nine rounds, one can get a nine round characteristic with probability $2^{-52.5}$. There are three similar characteristics that can be get utilizing this because the four S-boxes are the same.

Figure 3.3 is the best eight round characteristic for LOKI presented in [6]. The probability of this characteristic is 2^{-46}. The extension of this iterative characteristic to nine rounds does not reduce the probability. Using this characteristic it is possible to break eleven rounds variant of LOKI, even faster than exhaustive search. Examining some complementation properties of LOKI, it is shown that the complexities of certain attacks can be reduced.

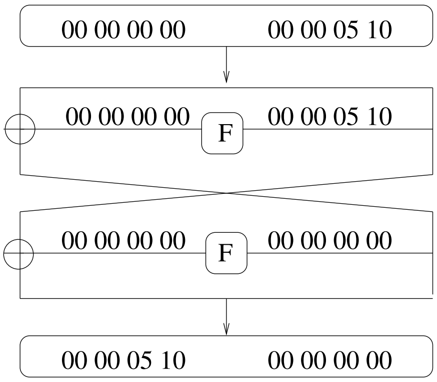


Figure 3.2: Two Round Differential Characteristic for LOKI

3.3 Differential Cryptanalysis of Lucifer

Lucifer, which is a precursor to DES, has a variety of published variants. The variant attacked in [6] is a substitution-permutation network, using two different S-boxes. In the round function of DES, the input value is first XORed with the key then, becomes an input for the S-boxes. Different from DES, in round function of Lucifer, input bits of the S-boxes are not XORed with the key, instead the key determines which S-box is placed in the round between the two different S-boxes. Although the particular S-boxes are not specified in the published paper of this variant of Lucifer, the attack is mounted using the third and fourth lines of S_1 of DES as the two S-boxes of Lucifer and it is stressed that other choices of the S-boxes give similar results. The attack based on the investigation of the equalities in the outputs of two S-boxes when a particular input is given. Some tables are constructed for this investigation. To illustrate, one of these tables shows the output bits that are equal for both S-boxes and another one shows output bits that are equal in pairs for either S-box.

The best attack to break eight round Lucifer in [6] uses 24 chosen plaintexts with 2^{21} operations. For the attack, the previously constructed tables are used. The plaintext differences are chosen to create 8 and A as an S-box input in the second round by the two members of the pairs. The certain bits of the subkeys in rounds six, seven and eight are determined.

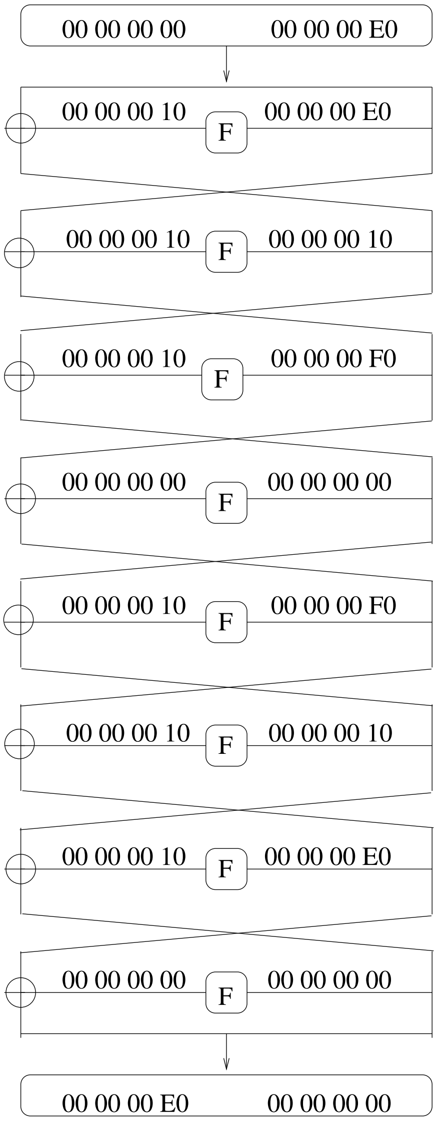


Figure 3.3: Eight Round Differential Characteristic for LOKI

3.4 Differential Cryptanalysis of RDES

RDES is derived from DES by replacing the deterministic swapping of the halves of the data between rounds in DES by a conditional swapping. The swap of the halves occurs only if a particular key bit has the value 1. This is the unique discrimination between DES and RDES. In [7], an improved version of differential cryptanalysis is introduced and applied to RDES. New concepts called conditional characteristics and key fraction of a conditional characteristic are used for the attack. The conditional characteristics can be defined roughly as the key dependent characteristics and the key fraction is defined as the ratio between the size of a subset of the key space corresponding to the conditional characteristic and the size of the key space.

These conditional differential cryptanalytic techniques are used to attack RDES. The following two observations show that many keys of RDES are quite weak. First observation is that one of every 2^{15} keys does not swap the data even once. Second observation is that one of every 2^{15} keys swaps the data just once before the last round. Now, using the conditional technique and the observations on the simple weaknesses on RDES it can be shown that almost any key of RDES is weaker than the corresponding key of DES.

For the attack of RDES the two round characteristic used for the attack of DES in [5] is used. It cannot be directly iterated but two 1-round characteristics of this two round characteristic can be combined. That is, there is always a way to combine the two 1-round characteristics of the two round characteristic shown in figure 3.4 for every choice of the swaps.

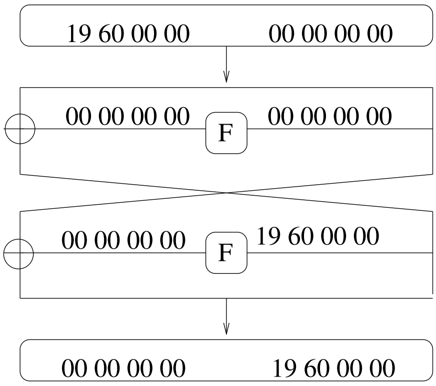


Figure 3.4: Characteristic for the Attack on RDES

Considering for iterating these two 1-round characteristics (the two rounds of the two round characteristic shown in figure 3.4) using conditional technique, it is concluded that the attacks on RDES are faster than the attacks on DES and require less chosen plaintexts in some cases. RDES is designed to be immune against differential cryptanalysis. However, it is concluded that RDES is not more secure than DES.

3.5 Differential Cryptanalysis of IDEA-X/2

IDEA, namely the International Data Encryption Standard, is derived from PES and designed for a replacement for DES by Xuejia Lai and James Massey. It is a 64 bit, substitution-permutation network block cipher using 128 bit key and running on 8.5 rounds. Its operations are bitwise exclusive or, addition modulo 2^{16} and multiplication modulo $2^{16}+1$. An illustration is given in figure 3.5.

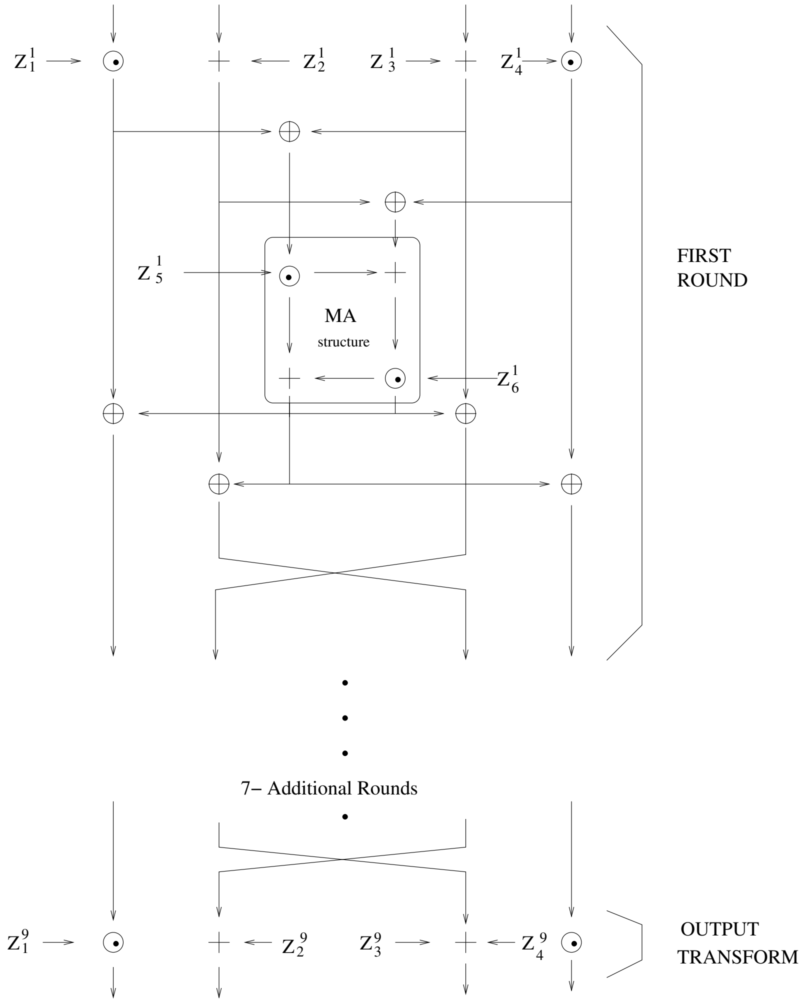


Figure 3.5: General View of IDEA

where Z_i^r denotes the subkey i used in round r. IDEA-X/2 is constructed by only modifying the addition modulo 2^{16}. In this modification, the addition modulo 2^{16} turns into XOR operation. So, Z_2^r and Z_3^r are inserted by XOR operations.

We will report the conclusions of the differential attack on IDEA-X/2 presented in [8]. We have used the same notations with [8] to stay away from any confusions. For the attack, some properties of the groups Z_2^{16} and $GF(2^{16}+1)^*$ and differential properties of a certain isomorphism, Φ, are investigated. The isomorphism, Φ is a map from $GF(2^{16}+1)^*$ to Z_2^{16} and defined as $\Phi(g^x) = x$.

An eight round differential characteristic with probability 2^{-32} is constructed by iterating the one round characteristic with both input difference and output difference $(\delta, \delta, \delta, \delta)$ where $\delta = FFFD$. Then, using the observations of Φ, the probability of the characteristic is increased to 2^{-30}.

3.6 Differential Cryptanalysis of LOKI91

LOKI91 is a DES-like iterated block cipher of 64 bits designed by making some changes on LOKI89 to obtain a more secure cipher. It is a Feistel structure block cipher running 16 rounds. The 32 bit input of each round is XORed with the subkey before being an input of the round function. In the round function, the 32 bit input is expanded to 48 bits and then, using four 12×8 S-boxes and applying a permutation 32 bit output of the round function is obtained.

In [9], it is shown that there is no characteristic with a high enough probability to mount a succesful differential cryptanalysis. Although, there are difference pairs occuring with high probabilities such as $132/4096$, the probability of the best one round characteristic with a nonzero input difference is $52/4096$ because the inputs to two neighbouring S-boxes are dependent due to the addition of the key to the input before expansion. It is concluded that to find a high probability 13 round characteristic, some rounds must have zero input and output difference. For the purpose of searching a high probability characteristic, a classification of rounds is made by the following definition.

Definition 3.6.1 *[9]*

1. *A round with number i is of type A if the rounds with number $i-1$ and $i+1$ are zero rounds.*
2. *The rounds with number i and $i+1$ are a pair of type B if the rounds with number $i-1$ and $i+2$ are zero rounds.*
3. *The rounds with number i, $i+1$, $i+2$ are a triple of type C if the rounds with number $i-1$ and $i+3$ are zero rounds.*

The highest probabilities for the occurances of a round of type A, a pair of rounds of type B and a triple of rounds of type C are computed. Then, using the results it is shown that there is no 13 round characteristic with probability higher than 2^{-63} exists.

3.7 Differential Cryptanalysis of PES

The block cipher PES, namely the Proposed Encryption Standard, is introduced at Eurocrypt in 1990 prior to the design of IDEA. It is a 64 bit iterated block cipher, running 8 rounds. Operations of PES are bitwise XOR, addition modulo 2^{16}, and multiplication modulo $2^{16} + 1$ denoted by $\oplus$, $+$, and $\odot$ respectively in figure 3.6.

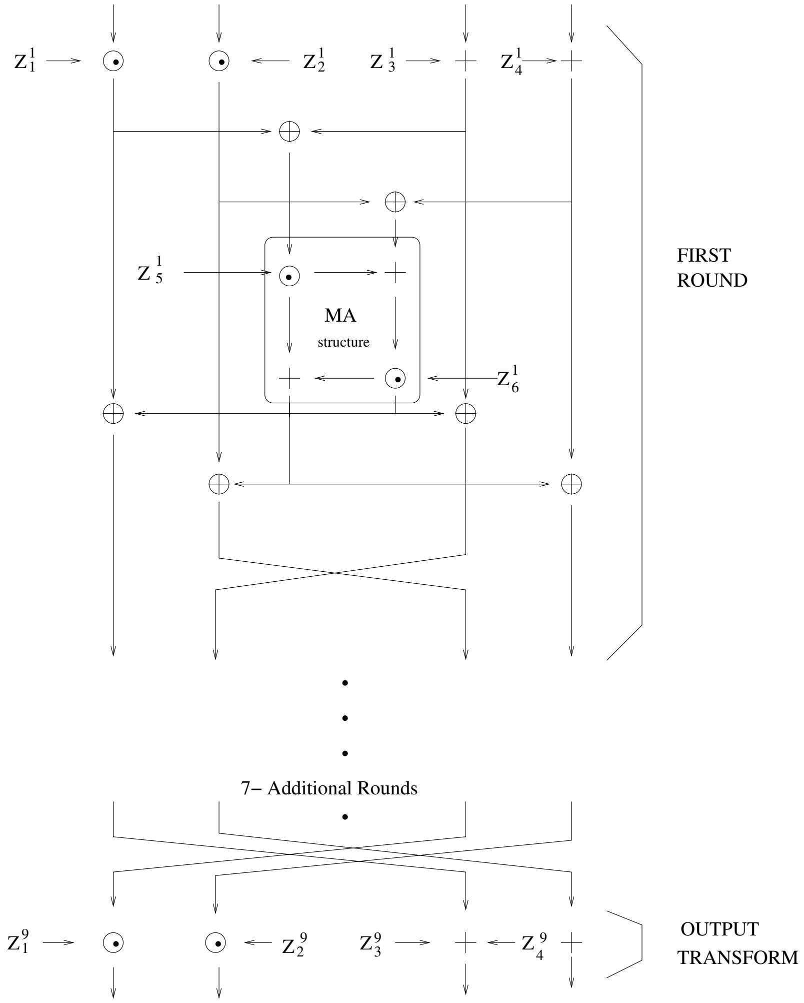


Figure 3.6: General View of PES

In [10], the concept of Markov ciphers and Markov chain which is used to determine whether a cipher is secure against differential cryptanalysis after sufficiently many rounds are introduced. This is a very significant concept in differential cryptanalysis. We will not mention about this but it is suggested for a reader to study. Instead, we will give a summary of the differential attack on PES presented also in [10].

ΔX, $\Delta Y(i)$, $P(v)$ are referred as the plaintext difference, the output difference of the i^{th} round and probability of discrete random variable v, respectively. Instead of characteristics, the notion of differentials is used in the paper because to mount a differential attack one does not concern about the value of the intermediate differences. The important thing is the knowledge of $\Delta Y(i-1)$ for determining the i^{th} round subkey. The most significant one round differentials are constructed using the eight plaintexts differences $(0, 0, 0, \gamma_i)$, where $i = 1, 2, ..., 8$, and $\gamma_1 = 2^{16} - 1$, $\gamma_2 = 1$, $\gamma_3 = 2^{16} - 3$, $\gamma_4 = 3$, $\gamma_5 = 2^{16} - 5$, $\gamma_6 = 5$, $\gamma_7 = 2^{16} - 7$, $\gamma_8 = 7$. Investigating each operation of PES in detail, the submatrix of the transition probability matrix of these values is approximated by the following matrix, T.

$$T = 10^{-7} \cdot \begin{pmatrix} 0 & 25460 & 12556 & 0 & 0 & 9417 & 698 & 0 \\ 25460 & 0 & 0 & 12556 & 9417 & 0 & 0 & 698 \\ 12556 & 0 & 0 & 0 & 6278 & 0 & 0 & 3139 \\ 0 & 12556 & 0 & 0 & 0 & 6278 & 3139 & 0 \\ 0 & 9417 & 6278 & 0 & 0 & 0 & 0 & 0 \\ 9417 & 0 & 0 & 6278 & 0 & 0 & 0 & 0 \\ 698 & 0 & 0 & 3139 & 0 & 0 & 0 & 0 \\ 0 & 698 & 3139 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Then, a lower bound on the probabilities of seven round differential (α^i, α^j), T^7 is found as;

$$T^7 = 2^{-58} \cdot \begin{pmatrix} 0 & 1.22 & 0.53 & 0 & 0 & 0.43 & 0.07 & 0 \\ 1.22 & 0 & 0 & 0.53 & 0.43 & 0 & 0 & 0.07 \\ 0.53 & 0 & 0 & 0.23 & 0.19 & 0 & 0 & 0.03 \\ 0 & 0.53 & 0.23 & 0 & 0 & 0.19 & 0.03 & 0 \\ 0 & 0.43 & 0.19 & 0 & 0 & 0.15 & 0.03 & 0 \\ 0.43 & 0 & 0 & 0.19 & 0.15 & 0 & 0 & 0.03 \\ 0.07 & 0 & 0 & 0.03 & 0.03 & 0 & 0 & 0 \\ 0 & 0.07 & 0.03 & 0 & 0 & 0.03 & 0 & 0 \end{pmatrix}$$

where the (i, j) entry in T^7 is $P(\Delta Y(7) = \alpha_j, \Delta Y(6) \in A, ..., \Delta Y(1) \in A, |\Delta X = \alpha_i)$ and

A=$\alpha_1, \alpha_2, ..., \alpha_8$ which is a lower bound on the (i, j) entry of Π^7. The best 7 round differential probability is approximately 1.22×2^{-58} belongs to the differential (α_1, α_2). The attack requires 2^{64} encryptions which specify the entire mapping from a plaintext to a ciphertext determined by the secret key. Hence, the attacker do not have to find the actual secret key.

3.8 Differential Cryptanalysis of the ICE Encryption Algorithm

We will summarize the differential attack on ICE presented in [3] and use the same notations with the paper. ICE is a 64 bit Feistel structure block cipher presented in 1997. It uses 64 bit key and four different S-boxes, denoted by S_0, S_1, S_2 and S_3. The round function of ICE, call F, takes 32 bit input and expand it to 40 bits using the expansion function. Before XORing this expanded data with the 40 bit subkey, a keyed permutation is performed. This swaps certain bits of the data belonging to S_0 and S_2 or S_1 and S_3 depending on the 20 bit subkey. Then, four 10×8 S-boxes, each of which takes 10 bit data, gives 32 bit input to the permutation function that does not depend on the key.

In [3], differential attacks on ICE and Thin-ICE, which is a fast variant of ICE, are presented. For the attack on ICE, the concept of conditional characteristics introduced in [7] and low Hamming weighted differences that adress only one S-box are used. A list of differences that can be used to construct a three round characteristic is given. The best three round characteristic is constructed from the list, to mount a six round attack. Then, it is enlarged to a four round characteristic which has the probability 2^{-13}. The characteristic is illustrated in figure 3.7. It is a conditional characteristic depending on bit 14 for the permutation subkey in round 2 and bit 2 for the permutation subkey in round 3. After examining the key scheduling algorithm, it is found that 20^{th} bit of the key must be equal to 1 and 12^{th} bit of the key must be 0 to make the characteristic valid for the attack.

Here $\alpha = 28$ and $\beta = 18$. 60 bits of the subkey are determined using this characteristic and remaining 4 bits are obtained by exhaustive search. The S/N for the attack is 2^{18}.

The attack can be extended up to 15 rounds by analysing some properties of the cipher because the straightforward extension is impractical after nine rounds. For instance, they observe that there are common bits in different S-boxes, and also between the first round and the last round subkey. The most significant observation is the realization of the improvement in the signal to noise ratio if one chooses to mount a 1R attack instead of 2R attack. Although this results

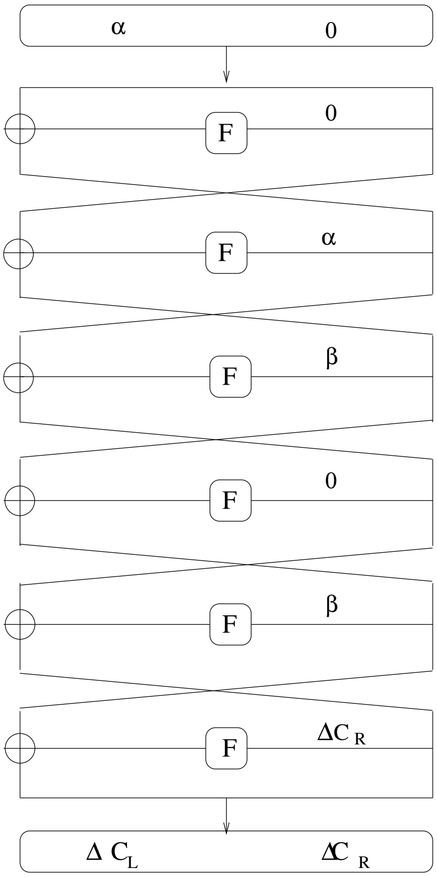


Figure 3.7: The Differential Characteristic for ICE

with a decrease in the probability of characteristic, it provides much more filtering and so helps to eliminate more wrong pairs. A 15-round attack requires at most 2^{56} plaintexts and the probability is approximately 2^{-52}.

3.9 Differential Cryptanalysis of a Reduced Round SC2000

SC2000 is a 128 bit block cipher running on six and a half rounds. Its structure consists of a combination of Feistel and SP network. It operates on 32 bit words and supports 128, 192 or 256 bit keys. We will describe two differential attacks on SC2000 presented in [12] and [13]. For not to cause any complications we will refer to the same notations used in these papers.

Description of SC2000

The round function of SC2000 comprises three functions namely, *I* where the data is XORed with the subkey, *B* which substitudes the data using an S-box namely S_4 and *R* which includes

F function. The following figure 3.8 is an illustration of F.

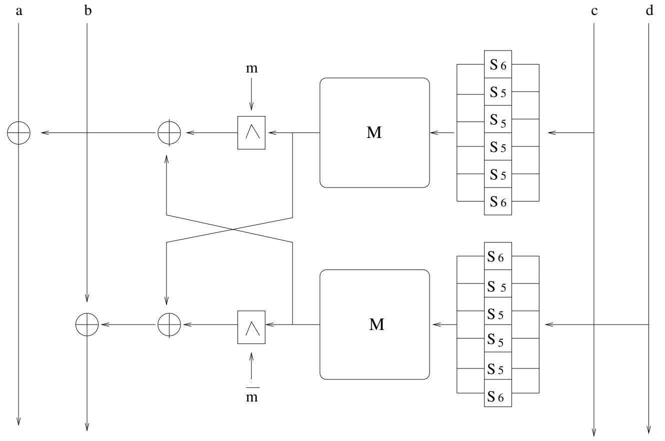


Figure 3.8: The Diagram of F

F consists of three functions S, M and L. S divides its 32 bit input into two groups of 6 bits and 4 groups of 5 bits. Then, each of the 6 bits data enter the S-box, S_6, and the 5 bits data enter the S-box, S_5. M is a linear function designed to ensure the diffusion. At this stage of the F function, the output of S is multiplied with a 32×32 matrix with elements from GF(2). Next, the data enters the last part of the F function, L. The two 32 bit data in two branches entering L are first masked with a constant which is 55555555 in odd rounds or 33333333 in even rounds considering the cipher begins with round 1. Then, they are XORed crosswisely with the unmasked 32 bit values. One round of the cipher is given in figure 3.9. Note that, the half round at the end of the cipher consists of *I-B-I*.

The Attacks

In [12], to find a high probability characteristic, they search for an iterative characteristics. For this purpose, the probabilities of the S-boxes are analysed. One of the best one round characteristics is given with both input value and output value equal to $(0, 00080008, 08090088, 0)$ and probability 2^{-33}. Several patterns of two round differential characteristics have been investigated. Among those, a pattern which is followed by the two round characteristic used for the attack and having the highest probability, 2^{-58}, is constructed by concatenating the patterns of differences shown in figure 3.10.

A differential attack on the following four and a half round SC2000 is presented.

$$I - B - I - R_5 \times R_5 - I - B - I - R_3 \times R_3 - I - B - I - R_5 \times R_5 - I - B - I - R_3 \times R_3 - I - B - I$$

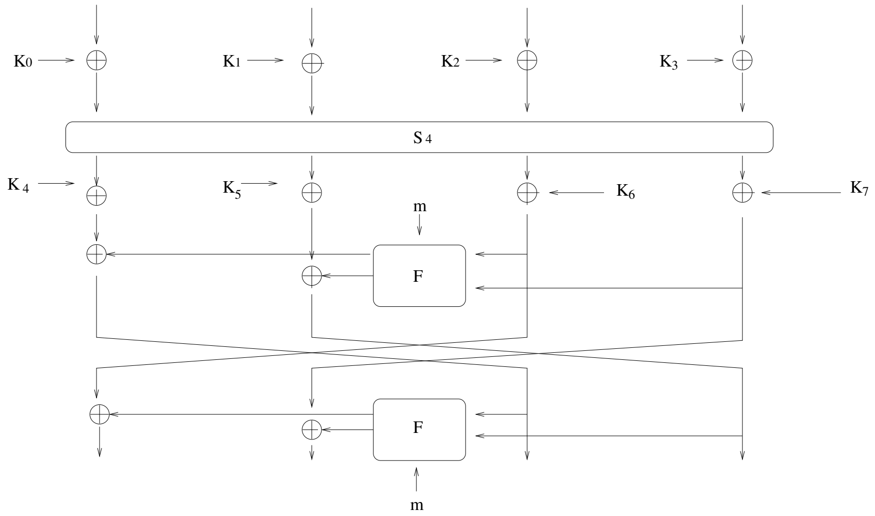


Figure 3.9: The round function of SC2000

For this attack, a three and a half round characteristic with probability 2^{-101} is built by concatenating the two round characteristic twice (an example of such a characteris is given in figure 3.11) and removing one *B* round.

The attack of the 4.5 round SC2000 requires 2^{104} encryptions and 2^{20} memory accesses. At the end, 40 subkey bits belonging to the first and the last *I* rounds are obtained [12].

Before this attack [12], in 2001, a differential attack on 4.5 round variant of SC2000 using 2^{110} chosen plaintexts is presented in [13]. 32 subkey bits both from the first and last round are obtained. In this paper, to create a differential characteristic, the first two layers of the *F* function are investigated.

Definition 3.9.1 *[13] Assume α is an n-bit string such that α=$\alpha_1, \alpha_2, ..., \alpha_n$. Support of α, denoted by $\chi(\alpha)$, is defined to be the set of coordinates where α has a nonzero value.*

$$\chi(\alpha) = i : \alpha_i \neq 0.$$

A search over the 32 bit words of output differences of *M* function, called ϵ, with Hamming weight six or less is made to determine an input difference of *F* function, δ, corresponding to ϵ with low Hamming weight satisfying $\chi(\epsilon) \subseteq \chi(\delta)$. Using this analysis, two 1 round characteristics beginning from the Feistel rounds are constructed. The first one, also shown

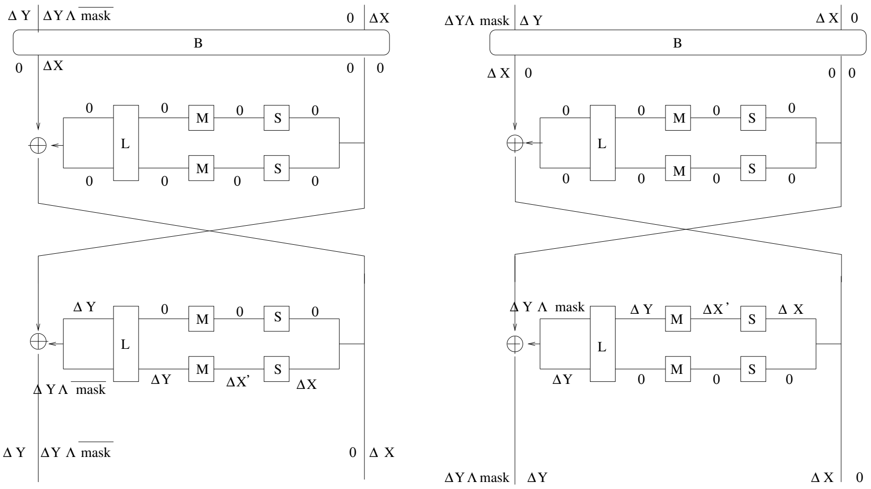


Figure 3.10: Patterns for SC2000

in figure 3.12 is $(\delta, 0, 0, 0) \rightarrow (0, \delta, 0, 0)$ with probability 2^{-30} and the other is $(0, \delta, 0, 0) \rightarrow (\delta, 0, 0, 0)$ with probability 2^{-29} where $\delta = 40220001$. Here, $x \rightarrow y$ means the input difference x causes the output difference y after one round with a certain probability.

By concatenating the one round characteristics, the following two 3.5 round differential charactristics are obtained:

- $(\delta, 0, 0, 0) \overset{F-F-S_4}{\rightarrow} (0, \delta, 0, 0) \overset{F-F-S_4}{\rightarrow} (\delta, 0, 0, 0) \overset{F-F-S_4}{\rightarrow} (0, \delta, 0, 0) \overset{F-F}{\rightarrow} (\epsilon = 40200000, 40000000, \delta, 0)$ with probability 2^{-107}
- $(0, \delta, 0, 0) \overset{F-F-S_4}{\rightarrow} (\delta, 0, 0, 0) \overset{F-F-S_4}{\rightarrow} (0, \delta, 0, 0) \overset{F-F-S_4}{\rightarrow} (\delta, 0, 0, 0) \overset{F-F}{\rightarrow} (00200000, \epsilon, \delta, 0)$ with probability 2^{-106}

Here, $x \overset{F}{\rightarrow} y$ means the input difference x causes the output difference y after F function with a certain probability. These characteristics start with the difference after the first S_4 layer. So, for using these characteristics structures are created. A pair of plaintexts which follows either of these two characteristics are called a right pair, while others are called wrong pairs.

3.10 Differential Cryptanalysis of Q

Q is a 128 bit block cipher with a variety of key sizes 128, 192 or 256 bits [18]. The layers in the round function uses similar structures to those used in Serpent and Rijndael. If the

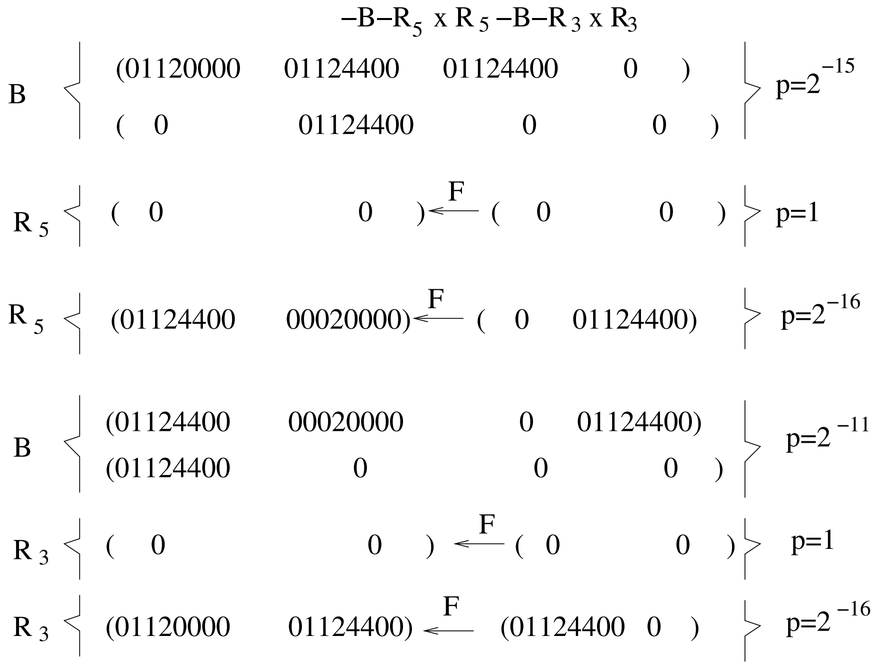


Figure 3.11: Two Round Characteristic

key size is 128 bits, the cipher consists of 8 rounds and for longer key sizes it has 9 rounds. The bits, bytes and words of nonzero differences are called active bits, active bytes and active words, respectively.

We will give a summary of the differential attack that is mounted on *Q* in [19]. The round function of *Q* includes two parts. The first part effects only the place of active bits, while the second part changes only which bytes in the row are active. Some tables that show the probabilities of given input differences with one or two active bits causing output differences with one or two active bits of the S-boxes in the layers byte substitution, named Bit-Slice S-box *A* and Bit-Slice S-box *B*, are constructed. These computations are done by ignoring the permutation layer for simplicity. After some foundations, the analysis is adapted to original *Q*. Using the tables, 2784 one round differentials are obtained. One of the best one round differential with probability $5 \cdot 2^{-20}$ is shown in figure 3.13 below.

The best differential probabilities up to 9 rounds are presented with a table. These are computed for *Q* without permutation layer. To illustrate the best probabilitiy for 8 round is $2^{-122.9}$ and for 9 rounds is $2^{137.9}$. The key recovery attack to 8 rounds uses the set of 7 round differentials starting with the second part of the first round. For the differentials, the input differences

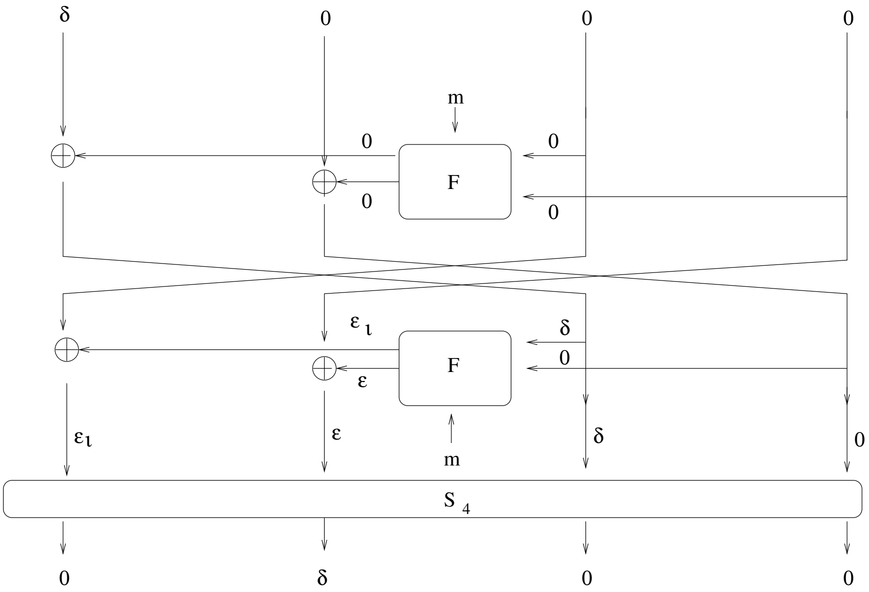


Figure 3.12: One Round Differential Characteristic for SC2000

$$\begin{bmatrix} 0 & \delta_i & 0 & \delta_i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow[\text{part1}]{2^{-14}} \begin{bmatrix} 0 & \delta_j & 0 & \delta_j \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow[\text{part2}]{5\text{x}2^{-6}} \begin{bmatrix} 0 & \delta_j & 0 & \delta_j \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3.13: One Round Differential

belong to the set

$$\begin{pmatrix} 0 & 0 & \delta_i & \delta_i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

where δ_i is the 8 bit value 2^i, $i \in 0, 1, ..., 7$ and the output differences belong to the set G that consists of differences with exactly two active bytes taking place in the same row and having the same differential δ_i for $i \in 0, 1, ...7$. The attack also uses structures and requires 2^{105} chosen plaintexts. The complexity is 2^{77}.

All the differentials over several rounds of Q described in [19] may be taken in the backward direction. The probabilities remain the same for both the forward or the backward direction. To attack on 9 rounds Q, the set of 8 round differentials starting from the second part of the ninth round in the backward direction is used. The input differences for differentials are from the set

$$\begin{pmatrix} 0 & \delta_i & 0 & \delta_i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

where δ_i is the same as it is defined before and the output differences are from the set G. The number of required chosen ciphertexts is 2^{125}. The complexity is 2^{128} and it can be improved to 2^{96} for 192 bit keys.

3.11 Differential Cryptanalysis of Nimbus

Nimbus is a 64-bit block cipher submitted for NESSIE project [20]. It uses 128 bit keys and runs five rounds. The round function includes the operations, bitwise XOR, multiplication modulo 2^{64} and a bit reversal function, g.

$$O_i = k^i_{odd} \cdot g(O_{i-1} \oplus k^i),$$

where O_i is the output of i^{th} round, k^i and k^i_{odd} which is always odd are subkeys of the i^{th} round.

A differential attack on full round Nimbus requiring 256 chosen plaintexts is presented in [21]. To obtain a one round iterative characteristic for the attack, investigations are made on the multiplication operation and two new differential properties of this operation are found. One round iterative characteristic has input and output differences as 01...10 with probability $1/2 + 2/2^{64}$. So, iterating this characteristic one obtains a five round characteristic with probability $(1/2 + 2/2^{64})^5 \cong 2^{-5}$. Also, it is noted that this probability can be increased to 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1} and even to 1.

CHAPTER 4

LINEAR CRYPTANALYSIS

4.1 Introduction to Linear Cryptanalysis

Linear cryptanalysis is a known plaintext attack introduced by M. Matsui [55] in 1993. The attack based on the observation of the statistical linear relations (approximations) between the plaintext, ciphertext and the key bits. Once an approximation with high probability is found, the attacker obtains an estimate of the parity bits of the key by counting on the parity bits of known plaintexts and ciphertexts. This kind of an approximation is in the following form:

$$P[i_1, i_2, ..., i_u] \oplus C[j_1, j_2, ..., j_v] = K[k_1, k_2, ..., k_w]$$

where $X[i_1, i_2, ..., i_t] = X[i_1] \oplus X[i_2] \oplus ... \oplus X[i_t]$ and $i_1, i_2, ..., i_u, j_1, j_2, ..., j_v, k_1, k_2, ..., k_w$ denote fixed bit locations. If we assume the fixed bit locations are chosen randomly, it is expected that the probability of the approximation is $1/2$. The deviation (or bias) from the probability $1/2$ is due to the poor randomization abilities of the cipher. Both being linear and affine are bad characterizations for a cipher. Therefore, for linear cryptanalysis, it is important to find a linear relation with a very high (or low) bias. To describe the linearity of the approximations in a simpler way, instead of the probability, the *bias* is considered. It is defined as $p - (1/2)$ where p is the probability.

To create a linear approximation for a cipher, at first, the linearization of the nonlinear operations are investigated. That is, linear approximations are derived for each component of the nonlinear operation by choosing a subset of the input bits and output bits. Assuming X is the input and Y is the output, a linear approximation of a nonlinear operation is of the form:

$$X[i_1, i_2, ..., i_u] \oplus Y[j_1, j_2, ..., j_v] = 0$$

where $i_1, i_2, ..., i_u, j_1, j_2, ..., j_v$ denote fixed bit locations. If a component is linear in the bits of the subset, then all the input values must have parity (exclusive-or) zero. Also, if a component is affine in the bits of the subset, then all the input values must have parity one. For each possible value of the inputs of the component, the parity of these bits are calculated. The approximation which has the highest magnitude of the bias, $|p - (1/2)|$, is the best one. It should be realized that, if the right hand side of the above approximation is one, then the bias will have the opposite sign.

A linear approximation for the whole cipher is constructed by combining the linear relations for each round logically. Also, the round linearization is based on the linearization of the nonlinear components. Therefore, to create a linear approximation for the whole cipher, the attacker should investigate the nonlinear components, at first. After finding an approximation for the nonlinear components, this is distrubuted to the rounds and then, to the whole cipher.

Before considering the details of linear cryptanalysis, note that we have changed some of the our notation just for the chapter of linear cryptanalysis. We start the numbering of the bits from right and accept the first bit as zero. Then, for this notation the permutation operation for the sample cipher is given in table 4.1. Also, note that X_i denotes the i^{th} round input and K_i denotes the i^{th} round key.

Table 4.1: Permutation Table for Linear Cryptanalysis of the Sample Cipher

input	7	6	5	4	3	2	1	0
output	1	2	0	3	6	7	4	5

4.1.1 Piling-up Lemma

A linear approximation with a high magnitude of the bias is constructed by investigating the linearization of the rounds. Round approximations are constructed by combining the linear relations of nonlinear components. The question is how can we compute the bias of the approximation for the whole cipher. Actually, the computation in a exhaustive way cannot be possible for the full-size ciphers, although it can be easily done for the nonlinear components. However, by making a number of assumptions, the bias can be approximated by using the bias of the nonlinear components. This method is the so-called Piling-up lemma found by Matsui:

Piling-up Lemma [55]: Let $X_1, X_2, ..., X_n$ be n independent random variables taking on values from $\{0, 1\}$. Assume the probability of X_i equals to zero is p_i. Then, the probability of the sum $X_1 \oplus X_2 \oplus ... \oplus X_n$ equals to zero is:

$$P(X_1 \oplus X_2 \oplus ... \oplus X_n = 0) = 1/2 + 2^{n-1} \prod_{i=1}^{n} (p_i - 1/2)$$

Proof: We will use the method of induction.

Step1(For n=1):

$$P(X_1 = 0) = (1/2) + 2^{1-1} \prod_{i=1}^{1} (p_i - 1/2) = (1/2) + (p_1 - 1/2) = p_1.$$

Step2(For n=2): Note that, $P(X_i = 1) = 1 - p_i$ since $P(X_i = 0) = p_i$. Then, we have $P(X_1 \oplus X_2 = 0) = P(X_1 = X_2) = P(X_1 = 0, X_2 = 0) + P(X_1 = 1, X_2 = 1) = p_1 p_2 + (1 - p_1)(1 - p_2) = p_1 p_2 + (1 - p_2 - p_1 + p_1 p_2) = 1 - p_1 - p_2 + 2p_1 p_2 = (1/2) + (1/2) + 2p_1 p_2 - p_1 - p_2 = 1/2 + 2(p_1 - 1/2)(p_2 - 1/2) = 1/2 + 2^{2-1}(p_1 - 1/2)(p_2 - 1/2)$, since X_1 and X_2 are independent.

Step3(For n=k-1): (Induction assumption) Assume, we have:

$$P(X_1 \oplus X_2 \oplus ... \oplus X_{k-1} = 0) = 1/2 + 2^{(k-1)-1} \prod_{i=1}^{k-1} (p_i - 1/2).$$

Step4(For n=k): $P(X_1 \oplus X_2 \oplus ... \oplus X_{k-1} \oplus X_k = 0) = P((X_1 \oplus X_2 \oplus ... \oplus X_{k-1}) \oplus X_k = 0) = (1/2) + 2 \cdot [(2^{(k-1)-1} \prod_{i=1}^{k-1}(p_i - 1/2)) \cdot (p_k - 1/2)] = (1/2) + 2^{k-1} \prod_{i=1}^{k}(p_i - 1/2))$ by using the induction assumption and step2.

The approximation of the whole cipher is a chain of connected linear approximations. As it is given in the hyphothesis of the lemma, it is assumed that the bias of each chain is independent. If this independence assumption is not fulfilled, then the computation of the bias can be different from the value that is predicted by the Piling-up lemma.

4.1.2 Linear Approximation Tables

The linear approximation table (LAT) of a nonlinear operation shows the number of zero parities for each of the possible subsets of the input and output bits of this operation. The division of a particular element of the table by 2^n indicates the bias of the approximation which consists the equality of the corresponding input and output parity where n denotes the number of input bits. Each row of LAT presents a particular parity of inputs, while each column presents a particular parity of the outputs.

Once linear approximations for the nonlinear operations are found, the attacker tries to built an approximation such that $P[i_1, i_2, ..., i_u] \oplus C[j_1, j_2, ..., j_v] = K[k_1, k_2, ..., k_w]$ for the whole cipher by using them. So, LAT is a very significant tool for linear cryptanalysis to find high probability approximations which is required for the attack.

However, it cannot be always possible to use it. To illustrate, LAT for the S-boxes of FEAL has 2^{24} entries and so, it is very difficult to analyse all these entries.

LAT can be constructed by the help of a computer but we will show how to construct it manually for the S-box of the sample cipher, because the input and output sizes are small. This way of construction is useful for a learner. Assume, we need to find the bias of the approximation $X_0 \oplus X_2 = Y_1 \oplus Y_2 \oplus Y_3$. To find the number of holds for this approximation we need to investigate all the 16 input values and their corresponding output values. This is shown in table 4.2.

Table 4.2: Some Parities of the S-box of the Sample Cipher

input	output	$X_2 \oplus X_0$	$Y_3 \oplus Y_2 \oplus Y_1$	$X_3 \oplus X_1$	$Y_2 \oplus Y_1 \oplus Y_0$
0000	1010	0	0	0	1
0001	0011	1	1	0	0
0010	1000	0	1	1	0
0011	0101	1	1	1	0
0100	1100	1	0	0	1
0101	0000	0	0	0	0
0110	0010	1	1	1	1
0111	1111	0	1	1	1
1000	0110	0	0	1	0
1001	1110	1	1	1	0
1010	0001	0	0	0	1
1011	0100	1	1	0	1
1100	1001	1	1	1	1
1101	0111	0	0	1	1
1110	1101	1	0	0	0
1111	1011	0	0	0	0

As it is seen from the table 4.2, $X_0 \oplus X_2$ equals to $Y_1 \oplus Y_2 \oplus Y_3$ exactly 12 times. Then, the probability of the approximation $X_0 \oplus X_2 = Y_1 \oplus Y_2 \oplus Y_3$ is $12/16$, since there are 16 input values. So, the bias is $(12/16) - (1/2) = 4/16$. For setting this bias on LAT, we note that the input parity $(X_0 \oplus X_2)$ is presented by 5 and the output parity $(Y_1 \oplus Y_2 \oplus Y_3)$ is presented by E. The element corresponding to this approximation is +4 since the bias is $4/16$. This can be seen from the table 4.3 (LAT) for the S-box of the sample cipher. All the other entries can be calculated in a similar way.

Table 4.3: Linear Approximation Table

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	+2	+2	−4	+2	0	0	−2	−2	0	−4	−2	0	−2	+2	0
2	0	+2	−2	0	0	−2	+2	0	0	−2	−2	+4	0	+2	+2	+4
3	0	0	0	0	−2	−2	−2	−2	−2	+2	+2	−2	−4	0	0	+4
4	0	+2	0	−2	0	−2	0	+2	+2	−4	+2	0	−2	−4	−2	0
5	0	0	−2	+2	+2	+2	+4	0	0	0	+2	−2	−2	−2	+4	0
6	0	0	−6	−2	0	0	−2	+2	−2	+2	0	0	+2	−2	0	0
7	0	+2	0	−2	−2	+4	−2	0	+4	+2	0	+2	−2	0	+2	0
8	0	+2	0	+2	+2	+4	−2	0	0	−2	0	−2	+2	0	−2	+4
9	0	+4	−2	+2	0	0	−2	−2	−2	−2	0	0	−2	+2	0	−4
A	0	0	+2	+2	+2	−2	−4	0	0	0	+2	+2	+2	−2	+4	0
B	0	−2	0	−2	−4	+2	0	−2	−2	−4	+2	0	+2	0	+2	0
C	0	−4	0	0	+2	+2	−2	+2	−2	−2	−2	+2	−4	0	0	0
D	0	+2	+2	0	0	+2	+2	0	−4	+2	+2	+4	0	−2	−2	0
E	0	−2	−2	0	+2	0	0	−6	+2	0	0	+2	0	−2	−2	0
F	0	0	0	−4	+4	0	0	0	0	0	+4	0	0	+4	0	0

One cannotice some similarities between the DDT (table 2.1) and LAT (table 4.3) of sample cipher:

1. Each element on the LAT is even.

2. The entries of the first row are all zero except the first entry. Each of these entries presents the bias of the approximations including a nonempty subset of output bits and an empty subset of input bits. Note that, any linear combination of output bits must have an equal number of zero's and one's for a bijective S-box. Hence, we have all zero's in the row corresponding to no set of input bits except the first entry.

3. The entries of the first column are all zero except the first entry. This is because of the same reason that we mentioned in property 2. One should note that this property is valid since the S-box is bijective.

4. The first entry is +8 because the probability of the approximation including no input bits and no output bits is 1.

5. The sum of the entries in each row equals to 2^n or -2^n.

6. The sum of the entries in each column equals to 2^n or -2^n.

4.1.3 Constructing One-Round Linear Characteristic

In [57], it is shown that the formalism of differential cryptanalysis can be adopted to linear cryptanalysis. Namely, characteristics can be defined, concatenated and used in a very similar manner as in differential cryptanalysis [57].

Although the characteristics of differential and linear cryptanalysis look so similar, there are so many significant discriminations between them. To illustrate, the bits that are set in linear characteristics indicate the subset of bits whose parity is approximated; that is, they do not indicate the bit differences as in differential characteristics.

A linear characteristic can be informally defined as a sequence that consists of the bit positions of plaintexts, ciphertexts, input and output datum of the rounds that appear in the linear approximation. The following is the formal definition of a one round linear characteristic. The definition of an n round characteristic will be given in the next section.

Definition 4.1.1 *([57]) Let Ω_P be the subset of bits of the data before the round, Ω_C be the subset of bits of the data after the round and Ω_K be the subset of bits of the key whose parity is approximated. A one round characteristic is a tuple (Ω_P,Ω_C,Ω_K,1/2+ϵ) such that $(\Omega_P)_L = (\Omega_C)_L = A$, $(\Omega_P)_R \oplus (\Omega_C)_R = a$ and $1/2 + \epsilon$ is the probability that a random block P and its one round encryption C under a random key K satisfies $P \cdot \Omega_P \oplus C \cdot \Omega_C \oplus K \cdot \Omega_K = 0$ where " $\cdot$ " denotes binary scalar product of two binary vectors.*

For linear cryptanalysis, the nonlinear operations of the round function are investigated, at first. Then, by using this investigation, linear approximations for the rounds are constructed. So, we will first find linear approximations for the only nonlinear operation of the sample cipher, namely the S-box by using table 4.3 (LAT). As it is seen from the table, the unique highest probability approximation for the S-box is the one including no set of input bits and output bits. So, the best approximation includes no active S-boxes. We have shown this in figure 4.1. Here, $(-)$ means parity of an empty subset and $[t_1, ..., t_n]$ are the subset of bits whose parity is approximated.

One should realize that in figure 4.1, parities of the input values and the output values of the XOR operation are the same. Also, the XOR of right half input value of plaintext and the

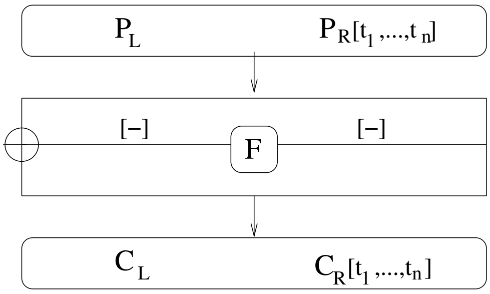


Figure 4.1: One Round Linear Characteristic

input value of the round function equals to the right half of the ciphertext. This is due to the reason that the bits set in the linear characteristic denote the positions of bits whose parity is approximated.

Another high probability one round characteristic is shown in figure 4.2. We have constructed this by examining the LAT. The highest magnitude of bias is belonging to the approximation on the table is −6 after 8. Although this is also the case when the input value is 14 and output value is 7, we have chosen the input value is 6 and output value is 2.

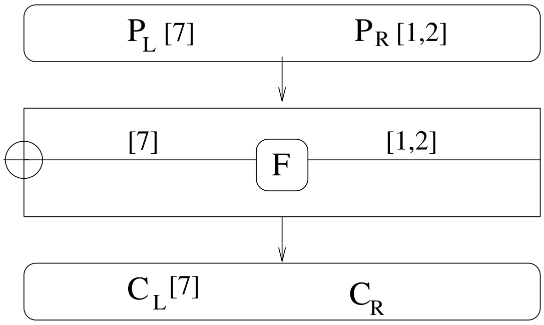
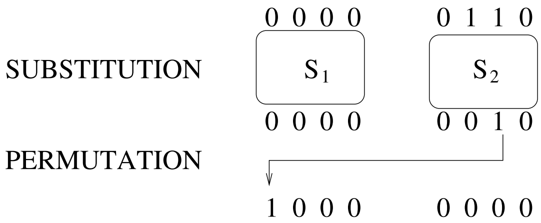


Figure 4.2: One Round Linear Characteristic for Sample Cipher

Observing the operations of the round function, this can be seen more clearly. The input parity to the S-boxes is 00000110 and the output parity is chosen as 00000010 with bias $-6/16$. Then, considering the permutation it can be seen that the output parity is 10000000. These are shown in the figure below.



4.1.4 Constructing Three-Round Linear Characteristics

In the previous section, we have shown how to construct linear approximations for S-boxes of the sample cipher and find a one round characteristic by using this. In [57], it is shown that the linear characteristics can be concatenated according to some rules. So, in this section we will concatenate one round characteristics to get a three round characteristic for sample cipher.

Similar to differential characteristics, a three round linear characteristic can be constructed by adding a characteristic with probability one between two identical high probability characteristics. To illustrate, for sample cipher a three round characteristic can be built by concatenating the characteristics shown in figure 4.1 and 4.2. This can be seen in figure 4.3. As we have mentioned before, it is important to realize that the right side of the input data of any round is the XOR of the input data of the round function and the data which becomes the left half of the next round. Also, input values and the output values of the XOR operations are the same. This is true because we do not denote the actual values of the bits in linear characeristics. We denote the positions of the bits instead.

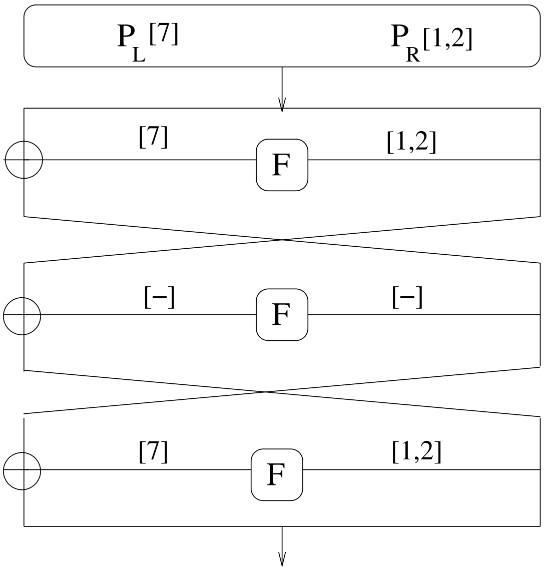


Figure 4.3: Three Round Linear Characteristic for Sample Cipher

The concatenation of the linear characteristics is formally defined. In this definition, we stick to the notations of the definition 2.1.1.

Definition 4.1.2 *([57]) An n round characteristic* $\Omega^1 = (\Omega^1_P, \Omega^1_C, \Omega^1_K, 1/2 + \epsilon_1)$ *can be con-*

catenated with an m round characteristic $\Omega^2 = (\Omega^2_P, \Omega^2_C, \Omega^2_K, 1/2 + \epsilon_2)$ *if* Ω^1_C *equals to the swapped value of the two halves* Ω^2_P*. The concatenation of the chracteristics* Ω^1 *and* Ω^2 *(if they can be concatenated) is the* $(n+m)$ *round characteristic* $\Omega = (\Omega^1_P, \Omega^2_C, \Omega^1_K, 1/2+2\cdot\epsilon_1\cdot\epsilon_2)$.

4.1.5 Adding One Round to the Three-Round Linear Characteristic

In this section, the three round characteristic shown in figure 4.3 will be expanded to four rounds. Namely, we will add one more round at the end of the three round characteristic. As shown in figure 4.4, the positions of the output bits of the round function in the fourth round are 1 and 2.

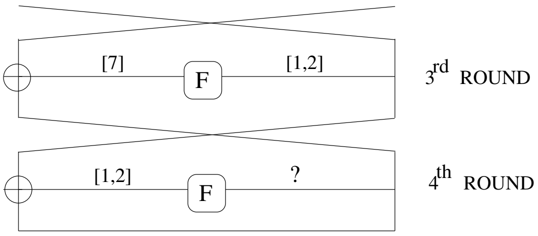


Figure 4.4: The Last Two Rounds of the Four Round Characteristic

A high probability input value should be determined for the fourth round. For this reason, we first find the output value of the S-boxes as 10010000 by going backwards through the permutation operation. Then, we have chosen the input value of the S-boxes as 01000000 which has probability $1/2 + (-4/16) = 1/4$ using LAT (table 4.3). The four round characteristic is shown in figure 4.5.

The probability of this characteristic can be computed using the piling-up lemma as $(1/2) + 2^{4-1}\cdot(-4/16)\cdot(-6/16)\cdot(-6/16) = 1/2-9/32$. This is the best four round linear characteristic for the sample cipher, but actually it is not unique. The attack can also be mounted succesfully with the other same probability characteristics.

At the end of the subsection 1.1.5 we have concluded some remarks about finding a differential characteristic. These remarks are all valid for the linear characteristics if we consider it is written LAT instead of DDT in these remarks.

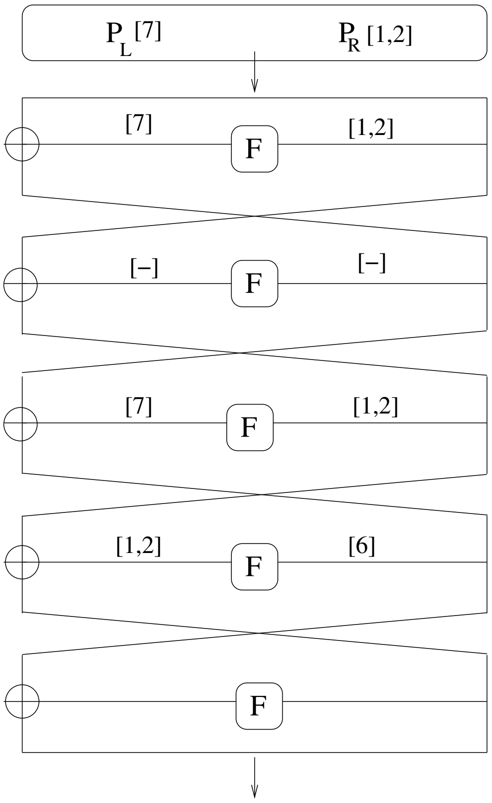


Figure 4.5: The Linear Characteristic for Sample Cipher

4.1.6 Linear Attack on Sample Cipher

In this section, we will mount a linear attack to the sample cipher using the four round characteristic shown in figure 4.5. First, we should extend the linear approximations of the round functions to the entire algorithm. Actually, one can obtain the following approximation by cancelling out the common terms in linear approximations of each round:

$$P_L[7] \oplus P_R[1,2] \oplus C_R[1,2] \oplus (f(C_R, K) \oplus C_L)[6,7] = K_1[1,2] \oplus K_3[1,2] \oplus K_4[6],$$

where K is the fifth round key. The right side of the equation, namely XOR of the subkey bits that are involved in the linear approximation is a constant, zero or one. The actual value of this constant can be determined while attacking.

When a high probability $n-1$ round linear characteristic is given for an n round DES-like cipher, a method which Matsui calls Algorithm 2 can be applied to obtain the subkey of the last round [55].

Algorithm 2 [55]: Assume for an n round DES-like cipher, a high probability $n-1$ round linear expression is given as

$$P[i_1, i_2, ..., i_u] \oplus C[j_1, j_2, ..., j_v] \oplus f(C_R, K)[t_1, t_2, ..., t_r] = \sum K_i$$

where K is the last round subkey and $\sum K_i$ is XOR of some bits of the intermediate round subkeys.

Step1: Let k_i be the candidate subkeys such that $0 \leq i \leq 2^n$ and $k_i = i$ in hexadecimal notation. For the each candidate subkey, let T_i be the number of plaintexts such that the left side of the $n-1$ round approximation is equal to zero.

Step2: Let T_{max} be the maximal value and T_{min} be the minimal value of all T_i's. It is clear that $|T_{max} - N/2|/N$ and $|T_{min} - N/2|/N$ indicates the magnitude of the bias when the subkey is corresponding to T_{max} and T_{min}, respectively.

- If $|T_{max} - N/2| > |T_{min} - N/2|$, then adopt the key candidate correponding to T_{max} and guess

$$K[k_1, k_2, ..., k_w] = \begin{cases} 0, & \text{if } p > 1/2 \\ 1, & \text{otherwise} \end{cases}$$

- If $|T_{max} - N/2| < |T_{min} - N/2|$, then adopt the key candidate correponding to T_{min} and guess

$$K[k_1, k_2, ..., k_w] = \begin{cases} 1, & \text{if } p > 1/2 \\ 0, & \text{otherwise} \end{cases}$$

By this algorithm, one can both guess the last round key bits and the value of the XOR of the intermediate round key bits that places in the right hand side of the equation, namely $\sum K_i$. The algorithm can be understandable in a better way while attacking, so we will use this algorithm to attack the sample cipher.

Consider the approximation for the characteristic given in figure 4.5:

$$P_L[7] \oplus P_R[1,2] \oplus C_R[1,2] \oplus (f(C_R, K) \oplus C_L)[6,7] = K_1[1,2] \oplus K_3[1,2] \oplus K_4[6] = \sum K_i(*)$$

Let α be the deviation from $1/2$ of the probability that the linear expression for the complete cipher holds. For this attack $\alpha = -9/32$, namely the bias of $(*)$. In his paper [52], Matsui shows that the number of known plaintexts required in a linear attack is proportional to $1/\alpha^2$. In practice, the number of plaintexts is chosen as $1/\alpha^2$ multiplied with a small constant, say c. For our attack, it is sufficient to choose $c = 4$. So, approximately $4 \cdot 2^4$ pairs will be enough to mount the attack. This is a known plaintext attack, so the attacker is assumed to have 2^6

known plaintext-ciphertext pairs.

In the approximation $(*)$, the only unknown value is K, noting that $\sum K_i$ is zero or one which will be determined during the attack. Going backwards through the permutation operation, the bit positions 6 and 7 of the round output corresponds to the bit positions 1 and 2 of the output value of S-boxes. So, we will try to guess the part of the fifth round subkey that is used in S_2, since the first and second bits are outputs of S_2. There are $2^4 = 16$ possible values for the 4 bits of S_2, so 16 candidate subkeys $k_1, k_2, ...k_{16}$ exists. For each candidate key, a count is kept. At the beginning, all counts indicate zero. Firstly, for k_1, take the first plaintext-ciphertext pair and check whether the equation $(*)$ is satisfied or not. If it is satisfied, then increment the count of k_1 by one. If it is not satisfied, do not increment the count and take the second pair to do the same procedure. For each value of k_i, for all i, check the satisfaction of the equation $(*)$ for all the pairs and keep the count of it. Then, divide this count by number of plaintext-ciphertext pairs. Let the division of the count by the number of plaintext-ciphertext pairs be called the key count. The keys with the key count which has the smallest distance from $1/2$ to bias will give us the correct bits of the last round subkey, K. That is, for the count of the correct key, the bias of $(*)$ is approximately equal to (count/number of pairs)-($1/2$). So, it is aimed to find a partial subkey which has the closest value of (count/number of pairs)-($1/2$) to the bias of our characteristic. Actually, the count of the correct key differs the greatest from half the number of plaintext/ciphertext samples among all the counts, because a wrong key is assumed to result in a relatively random guess at the bits entering the S-boxes of the last round.

For determining $\sum K_i$, one investigates the sign of the (count/number of pairs)-($1/2$). The sign of the bias of $(*)$ is negative, as we found in subsection 2.1.5. We have chosen the key whose count gives the greatest value for (count/number of pairs)-($1/2$). If this value is positive, we will choose $\sum K_i$ as 1 and If this value is negative, then we will choose $\sum K_i$ as 0.

4.2 Applications to Main Cryptographic Structures

In this section, we will give illustrations of some published papers for linear cryptanalysis on DES, RC5. Linear cryptanalysis is generally applied more succesfully on DES-like ciphers. This is due to the reason that the analysis of S-boxes can be done more easily. Also, the approximations found for the S-boxes can be easily distributed to the rounds [54]. However, this is generally more difficult for the ciphers which are not DES-like. This can be observed by studying the illustrations for linear attacks on RC5 [53, 54] in the following subsections.

4.2.1 Linear Cryptanalysis Method for DES Cipher

Linear Cryptanalysis is first introduced in [52] by M. Matsui. In this paper, it is shown that DES up to 16 rounds can be breakable by linear cryptanalysis faster than an exhaustive search for 56 key bits. Since the only nonlinear part of the round function is the S-box for DES, linear approximations of this component are studied. By searching for the greatest number on LAT for S-boxes, it is found that the best approximation for the round function of DES is:

$$x[15] \oplus F(x, K)[7, 18, 24, 29] = K[22], \text{ with probability } 0.19.$$

Here x is the input of F. By applying this approximation to the first and third rounds, the best three round linear characteristic is constructed with the expression $P_L[7, 18, 24, 29] \oplus C_L[7, 18, 24, 29] \oplus P_R[15] \oplus C_R[15] = K_1[22] \oplus K_2[22]$. It is shown in figure 4.6.

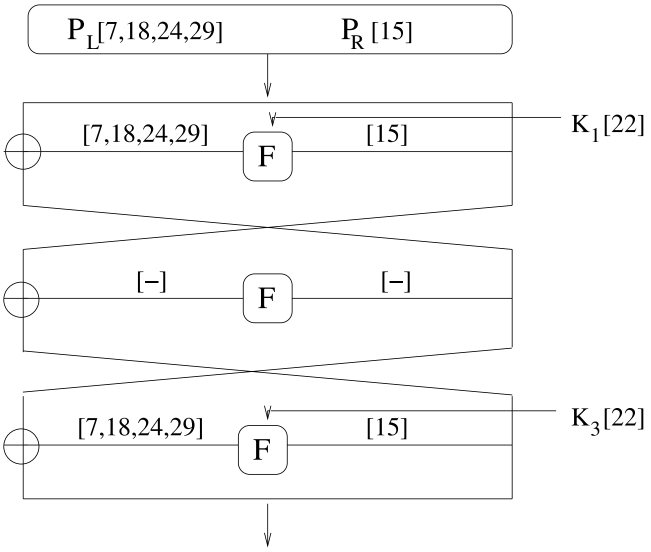


Figure 4.6: Three Round Linear Characteristic for DES Cipher

The probability of this three round characteristic is computed as $(12/64)^2+(1-12/64)^2 = 0.70$ by piling up lemma. This characteristic can be extended to five rounds by adding one round to the beginning and one round to the end as seen from figure 4.7. The probability of this five round characteristic can be calculated by piling up lemma as $(1/2) + 2^3 \cdot (-10/64)^2 \cdot$

$(-20/64)^2 = 0.519$.

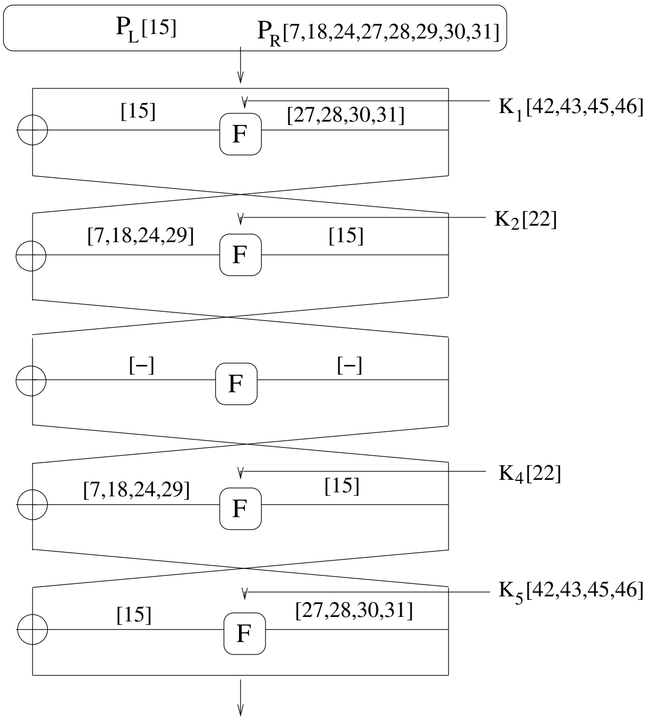


Figure 4.7: Five Round Linear Characteristic for Sample Cipher

The approximation for this five round characateristic can be easily calculated as:

$$P_L[15] \oplus P_R[7,18,24,27,28,29,30,31] \oplus C_L[15] \oplus C_R[7,18,24,27,28,29,30,31] = K_1[42,43,45,46] \oplus K_2[22] \oplus K_4[22] \oplus K_5[42,43,45,46].$$

The right side of the equation can be guessed by using approximately 2800 known plaintexts. These two characteristics are are stated as the best expression and the best probability characteristics given in [52]. The approximation which can break 16 rounds DES using 2^{47} known-plaintexts is the following:

$$P_L[7,18,24] \oplus P_R[12,16] \oplus C_L[15] \oplus C_R[7,18,24,29] \oplus F_{16}(C_R, K_{16})[15] = K_1[19,23] \oplus K_3[22] \oplus K_4[44] \oplus K_5[22] \oplus K_7[22] \oplus K_8[44] \oplus K_9[22] \oplus K_{11}[22] \oplus K_{12}[44] \oplus K_{13}[22] \oplus K_{15}[22]$$

Besides the given approximations, an attack on eight round DES is presented. For the attack, this approximation with probability $(1/2) + 1.95 \times 2^{-10}$ is used:

$$P_L[7,18,24] \oplus P_R[12,16] \oplus C_L[15] \oplus C_R[7,18,24,29] \oplus F_8(C_R, K_8)[15] =$$
$$K_1[19,23] \oplus K_3[22] \oplus K_4[44] \oplus K_5[22] \oplus K_7[22]$$

According to the approximation, it is seen that only one S-box is influenced, and so, six bits of the eighth round subkey can be determined. Applying this method to the first round, 14 subkey bits are obtained. The remainning key bits are derived exhaustively. The whole key is determined, in 20 seconds using approximately 2^{20} known plaintexts and in 40 seconds using 2^{21} known plaintexts.

Moreover, an only ciphertext attack of DES cipher is presented. In this case, the plaintexts are assumed to be nonrandom. Namely, if the probability of XOR of the plaintext bits in the approximation equals to zero is known to be different than $1/2$, then the plaintext bits can be eliminated from the approximation. So, we will be only left with the ciphertext bits and the subkey bits. Based on this idea, it is shown that if the plaintexts consist of natural English sentences represented by ASCII codes, then eight round DES can be breakable by using 2^{29} ciphertexts only. Also, if plaintexts consist of random ASCII codes, then it is shown that the eight round DES can be breakable with 2^{37} ciphertexts.

4.2.2 Linear Cryptanalysis of RC5 Encryption Algorithm

RC5 is a Feistel type block cipher designed by Rivest. It's block size (32,64 or 128), number of rounds (0 to 255) and key length (0 to 2040) are all variable. The first linear cryptanalytic attack mounted on RC5 is described in [53]. However, it is shown by Selçuk [54] that the attack does not attain the mentioned success rate due to some hidden assumptions. Also, the rounds are accepted to be independent and piling-up lemma is used for the computation of the probabilities. Nevertheless, the rounds are dependent in which case the piling-up lemma cannot be applied. Anyhow, we will give a summary and explain the complicated parts of this paper because it is an instructive paper and can be useful to learn some common mistakes done in linear cryptanalysis.

The Analysis of the Operations

There are three operations in RC5: The exclusive-or operation, the rotation operation and the addition operation. The rotation operation is data dependent. In figure 4.8, one can see one round (two half rounds) of RC5.

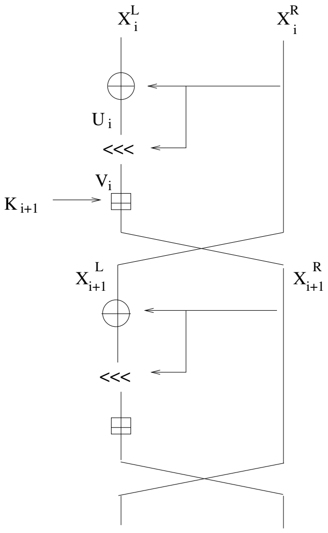


Figure 4.8: One Round of the RC5 Block Cipher

Now, considering the operations individually, and connecting logically, one can construct a linear approximation for a half round.

The exclusive-or operation is linear, so for this operation every approximation involving the same bits of X_i^L, X_{i+1}^L and U_i works with bias $\mp(1/2)$.

For the addition operation, we consider the approximation: X_{i+1}^R=V_i+K_{i+1}. If there was no consideration of a carry bit, this approximation would behave like an exclusive-or operation which is linear. So, we should be interested in the bits for which the addition behaves like an XOR operation. This is only happening for the least significant bit, since there comes no carry bit. So, the best linear approximation for the addition operation is $X_{i+1}^R[0]$=$V_i[0]$+$K_{i+1}[0]$ which holds with bias $\mp(1/2)$ for any K_i.

Lastly, we will analyze the rotation operation. In the paper, searching linear approximations for V_i=U_i <<< X_i^R is investigated in two cases: The linear approximations involving no bits of X_i^R (1) and the linear approximations involving $X_i^R[0]$ (2).

1. The linear approximations involving no bits of X_i^R are in the form

$$\sum_{j\in 0,1,...2^{w-1}} a_j V_i[j] = \sum_{j\in 0,1,...2^{w-1}} b_j U_i[j]$$

where a_j and $b_j \in \{0, 1\}$. For finding the probability of this equality, first of all consider the approximation involving just one bit of V_i and U_i. Assume it is known that,

$V_i[j]{=}U_i[t]$ for some determined j and $t \in \left\{0, 1, ..., 2^{w-1}\right\}$. Then, if the rotation amount is $t - j$, it is guaranteed that the linear approximation will be hold. However, for the other $w - 1$ rotations, the probability of linear approximation is $1/2$ since we do not know the equality of the bits. So, the probability of a linear approximation involving just one bit of V_i and U_i is $((w-1)/w) \cdot (1/2) + (1/w) \cdot 1 = 1/2 + 1/2w$. This case can be easily . An approximation involving 2^t bits of V_i and 2^t bits of U_i holds with probability $1/2 + 2^t/2w$.

2. The linear approximations involving $X_i^R[0]$ are in the form

$$V_i[0]{=}U_i[0] \oplus X_i^R[0]$$

The least significant bit is considered in the approximation because the output of the rotation operation is the input of the addition operation. So, it is necessary for combining the linear approximations of the two operations. Now, let L denote the event that the linear approximation, $V_i[0]{=}U_i[0] \oplus X_i^R[0]$, holds. Then, the probability of the approximation $P(L)$ can be calculated as;

$$P(L) = P(L|X_{i-1}^R modw = 0) \cdot P(X_{i-1}^R modw = 0) + P(L|X_{i-1}^R modw \neq 0) \cdot P(X_{i-1}^R modw \neq 0)$$

If there is no rotation (this is true with probability $1/w$), then $V_i[0]{=}U_i[0]$ and the approximation holds. If there is a rotation (this is true with probability $(w-1)/w$), then the approximation holds with bias $1/2$. Therefore, we have the probability:

$$1/w \cdot 1 + (1/2) \cdot ((w-1)/w) = (1/2w) + (1/2)$$

Constructing Linear Approximations

The approximation used for the attack is a one-bit linear approximation in [53]. We have previously seen that the addition operation behaves completely linear for the least significant bit. So, the attackers take the advantage of this property. First of all, they need to construct a half round linear approximation, after that they combine these approximations. For the addition operation it has been noted that the approximation $X_{i+1}^R[0]{=}V_i[0]{+}K_{i+1}[0]$ is the best one. This is joined with $V_i[0]{=}U_i[0] \oplus X_i^R[0]$. So, combining the three linear approximations for the individual operations, the following half round approximation can be obtained:

$$X_i^R[0] = X_{i-1}^L[0] \oplus K_i[0]$$

The probability of this approximation is $(1/2)+(1/2w)$ since the addition and the exclusive-or operations are linear for the least significant bit. This approximaton is denoted by E.
For constructing a linear approximation for more rounds, the first half round is needed to be considered. Since in the first round only the addition operation is used, we have the following approximations working with probability one:

$$X_1^L[0] = X_0^L[0] \oplus K_i[0] \text{ and } X_1^R[0] = X_0^R[0] \oplus S_1[0].$$

These are denoted by C and D respectively. Also, note that the trivial approximation coming from the definition of the cipher: $X_i^L[0] = X_{i-1}^R[0]$ is denoted by '-'.
From these, it is easy to see that $D - E - E - ... - E-$ is a linear approximation for i half rounds for i even and $CE - E - ... - E-$ is a linear approximation for i half rounds for i odd. For the attack the approximation $D - E - E - ... - E-$ for $2r$ half rounds is used and this can be written in the following form:

$$X_0^R[0] \oplus X_{2r}^L[0] = K_1[0] \oplus K_3[0] \oplus K_3[0] \oplus ... \oplus K_{2r-1}[0] = T$$

The right hand side of the equation is denoted as T. It is known that D and $-$ works with probability one. So, only E has an effect on the probability of the approximation. It is computed in [53] that, since E appears $r-1$ times, by piling-up lemma the approximation, $X_0^R[0] \oplus X_{2r}^L[0] = T$, holds with probability $(1/2) + (1/2w^{r-1})$. However, the piling-up lemma cannot be applicable because two consecutive half round approximations $X_i^R[0] = X_{i-1}^L[0] \oplus K_i[0]$ and $X_{i+2}^R[0] = X_{i+1}^L[0] \oplus K_{i+2}[0]$ are not independent [53]. Therefore, an attacker should be careful when applying a method developed for a specific cipher to a cipher of different type.

The Attack

In the attack, it is aimed to compute K_{2r+1} using the $2r$ approximation, $X_0^R[0] \oplus X_{2r}^L[0] = T$, in three steps. However, there is an important problem of the attack algorithm. At each step, $X_{2r+1}^L mod w$ is fixed to a certain value. This assumption leads to failures for computation of the probability in certain rounds and so, the success rate of the attack has changed.
It is proposed that, the subkey is obtained with the success rate $95 - 99\%$ [53]. However, Selçuk [54] implemented the attack on RC5 having word size 16 and 2 rounds with $2|p-1/2|^{-2}$ plaintext/ciphertext pairs for each different value of $X_{2r+1}^L mod w$ and observed the success rate

was around $11 - 15\%$. Also, he noted that there is no inrease in the success rate even if the data amount is increased.

The attack procedure uses three steps and finds each bit of K_{2+1} one by one:

Step 1: In this step, it is aimed to compute $K_{2+1}[0]$ by using N plaintext-ciphertext pairs satisfying $X^L_{2r+1} mod w{=}1$. Then, the one of the following approximations ((i) or (ii)) works with probability one. This can be more understandble from figure 4.9.

$$(i) X^L_{2r}[0] = X^L_{2r+1}[0] \oplus X^R_{2r+1}[1] \text{ if } K_{2r+1}[0] = 0$$
$$(ii) X^L_{2r}[0] = X^L_{2r+1}[0] \oplus X^R_{2r+1}[1] \oplus X^R_{2r+1}[0] \text{ if } K_{2r+1}[0] = 1$$

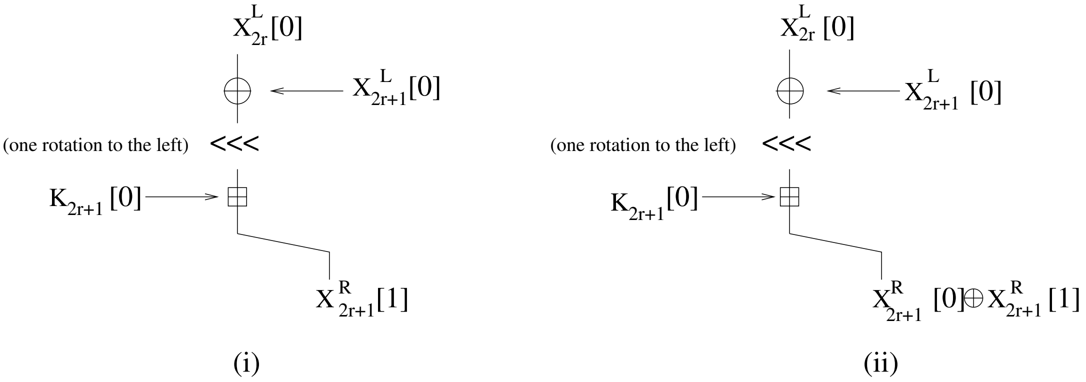


Figure 4.9: The Approximations with Probability 1 for RC5

Realize that, (i) has zero bias if $K_{2r+1}[0] = 1$ and also, (ii) has zero bias if $K_{2r+1}[0] = 0$. Now, putting the actual values of $X^L_{2r}[0]$ that are found (i) and (ii) into $X^R_0[0] \oplus X^L_{2r}[0]$, we get the following two quantities:

$$X^R_0[0] \oplus (X^L_{2r+1}[0] \oplus X^R_{2r+1}[1]) \text{ and } X^R_0[0] \oplus (X^L_{2r+1}[0] \oplus X^R_{2r+1}[1] \oplus X^R_{2r+1}[0])$$

Now, let Q_0 be the number of plaintexts such that the first quantity is zero and Q_1 be the number of plaintexts such that the second quantity is zero. Then we predict;

$$K_{2r+1}[0] = \begin{cases} 0, & \text{if } |Q_0 - N/2| \geq |Q_1 - N/2| \\ 1, & \text{otherwise.} \end{cases}$$

Step 2: In this step, it is aimed to compute T by using N plaintext-ciphertext pairs satisfying $X^L_{2r+1} mod w{=}1$ for given $K_{2r+1}[0]$. Consider $X^R_0[0] \oplus X^L_{2r}[0] = T$. In this equation, $X^R_0[0]$ is known. So, if we write something known instead of $X^L_{2r}[0]$, T can be obtained. The following approximation, also illustrated in figure 4.10, holds with probability one.

$$X^L_{2r}[0] = X^L_{2r+1}[0] \oplus X^R_{2r+1}[0] \oplus K_{2r+1}[0]$$

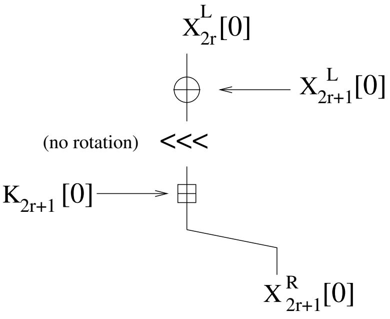


Figure 4.10: An Approximation with Probability 1 for RC5

Also, notice that the right hand side of the approximation is known. Let Q be the number of plaintexts such that

$$X^R_0[0] \oplus (X^L_{2r+1}[0] \oplus X^R_{2r+1}[0] \oplus K_{2r+1}[0])$$

is zero. Then, we predict;

$$T = \begin{cases} 0, & \text{if } Q \geq N/2 \\ 1, & \text{otherwise.} \end{cases}$$

Step 3: In this step, it is aimed to compute $K_{2r+1}[s]$ by using N plaintext-ciphertext pairs satisfying $X^L_{2r+1} mod w = s$ for given T and $K_{2r+1}[s-1...0]$, where $s = 1, ..., w-1$. Let $carry(s)$ denote the carry out from $V_{2r+1}[s-1...0] + K_{2r+1}[s-1...0]$. Then, by the help of figure 4.11, one can see that the following approximation holds with probability one:

$$X^L_{2r}[0] = X^L_{2r+1}[0] \oplus X^R_{2r+1}[s] \oplus K_{2r+1}[s] \oplus carry(s)$$

By using this approximation and $X^L_{2r}[0] = X^R_0[0] \oplus T$ we have;

$$K_{2r+1}[s] = (X^R_0[0] \oplus T) \oplus (X^L_{2r+1}[0] \oplus X^R_{2r+1}[s] \oplus carry(s))$$

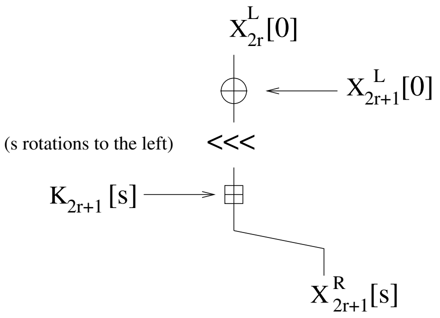


Figure 4.11: An Approximation with Probability 1 for RC5

Now, let Q be the number of plaintexts such that the above equation equals to zero. Then, we predict;

$$K_{2r+1}[s] = \begin{cases} 0, & \text{if } Q \geq N/2 \\ 1, & \text{otherwise.} \end{cases}$$

The Hiddden Assumptions

Selçuk [54], has shown that the attack in [53] does not work as expected. Actually, the important point of the attack is that, at each step the value of $X^{L}_{2r+1} mod w$ is fixed to a certain value. This leads to another two assumptions which causes to the failures in the probability computation:

- First Assumption: The probability $P(X^{R}_{i-1} mod w = 0)$ does not depend on $X^{L}_{2r+1} mod w$.
- Second Assumption: When $X^{R}_{i-1} mod w \neq 0$, the probability of the approximation $X^{R}_{i}[0] = X^{L}_{i-1}[0] \oplus K_i[0]$, does not depend on $X^{L}_{2r+1} mod w$.

For the detailed explanations for these assumptions, please refer to the paper [54].

4.2.3 Other Linear Attacks on RC5

The linear attack on RC5 proposed in [53] does not work as expected. Selçuk explained this in detail [54].

For the attack [54], the approximation, $X^{R}_{0}[0] \oplus (X^{R}_{2r+1} - K_{2r+1})[X^{L}_{2r+1} mod w] \oplus X^{L}_{2r+1}[0]$ is used. This is the same approximation used in [53]. Then, using the *1R-method* of Matsui, (Algorithm 2 in [55]) a linear attack is mounted.

However, since the piling-up lemma cannot be applied, the exact probability of the approximation and therefore, the success rates of the attacks based on this approximation cannot be computed. So, it can be understandable that the classical method of linear cryptanalysis does not work on RC5. That method is genearlly more applicable on DES-like ciphers.

The resistance of RC5 against linear cryptanalysis has been evaluated [56]. In this paper, different from the classical method of linear cryptanalysis, techniques related to the use of multiple linear approximations are used and the advantage of linear hull effect is taken. By this method, it is shown that RC5 with block size 128 can be broken up to 15 rounds.

4.3 Generalizations of Linear Cryptanalysis

Linear cryptanalysis is a very significant method of an attack for block ciphers. However, it requires large quantity of known plaintexts for several block ciphers, even for DES. Hence, some studies have been done to obtain extended versions. In this section, some of these extended versions of basic linear cryptanalysis, namely generalized linear cryptanalysis [59], linear cryptanalysis using multiple approximations [58] and partitioning cryptanalysis [60] will be studied.

4.3.1 Generalized Linear Cryptanalysis

In this type of extention, the linear expressions in basic linear cryptanalysis are generalized. These generalized linear expressions are called the I/O sums. We will briefly explain the building blocks of this type of the attack introduced in [59].

Definition 4.3.1 *An I/O sum S^i for the i^{th} round is the XOR of a balanced binary-valued function f_i (input function) of the round input X_i and balanced binary-valued function g_i (output function) of the round output Y_i. That is,*

$$S^i := f_i(X_i) \oplus g_i(Y_i)$$

I/O sums for successive rounds are linked if the output function of each round before the last coincides with the input function of the following round. (i.e. $f_i = g_{i-1}$)

Definition 4.3.2 *A multi-round I/O sum, $S^{(1...r)}$, is the sum when r successive S^i are linked. That is, $S^{(1...r)} := S^1 \oplus S^2 \oplus ...S^r = g_0(P) \oplus g_r(C)$.*

Two measures for usefulness of I/O sum are introduced, namely key-dependent imbalance and average-key imbalance. These measures correspond to the bias in basic linear cryptanalysis. To define these measures, we first give the definition of imbalance.

Definition 4.3.3 *Let V be a binary valued random variable taking values 0 and 1. The imbalance $I(V)$ is the non-negative real number defined by $|2P[V = 0] - 1|$.*

Realize that, the imbalance takes values between 0 and 1. The definitions of the two measures are as follows.

Definition 4.3.4 *The key-dependent imbalance of the I/O sum $S^{(1...r)}$ is the imbalance of this sum when the tuple of the round keys from first to r^{th} round are fixed.*

Definition 4.3.5 *The avarage-key imbalance of the sum $S^{(1...r)}$ is the expectation of the key-dependent imbalances.*
For an efficient attack, the I/O sum that is used for the attack should has a large avarage-key imbalance. Especially, if it is 1 (maximum) then, the I/O sum is called guaranteed.

The Attack Procedure

Generalized linear cryptanalysis exploits an effective $r - 1$ round I/O sum to obtain some portion of the last round key. The attack is similar to the basic linear attack and it proceeds as follows:

1. For each possible last round key K_r^i, set a counter $c[K_r^i]$ indicating zero.

2. Choose a plaintext-ciphertext pair, (P, C).

3. For each K_r^i, compute the decryption of C for one round. That is, evaluate $y_{r-1}^i := F_{K_r^i}^{-1}(C)$. Then, check if $g_0(P) \oplus g_{r-1}(y_{r-1}^i)$ equals to zero. If it does, then increment $c[K_r^i]$ by one.

4. Repeat two previous steps for all the N plaintext-ciphertext pairs.

5. Output all the keys K_r^i that maximize $|c[K_r^i] - N/2|$ as candidates of the last round key.

This attack does not give the actual key directly. It outputs key classes containing equivalent keys, where two keys k, k' are said to be equivalent if $g_{r-1}(F_{k'}^{-1}(Y))=g_{r-1}(F_k^{-1}(Y)) \oplus c$. This is because of the reason that the attack cannot distinugish between the equivalent keys. So, the outputs are key classes, each with representative keys. The key class containing the right key is called the right class.

Obviously, it is desirable that the output list contains only the right class. So, the success of the attack depends on the probability that the right class is included in the output list as one alone. This probability is called the success probability. It depends on the avarage key imbalance.

For an efficient attack, it is important to find effective I/O sums. This can be achieved by using the notion of ”*threefold sums*”. Details can be found in [59].

Generalized linear attack is applied to DES and IDEA but they are not so successful [59]. Also, one can study an application of this attack on SAFER in [62]. Although, it is found that SAFER is secure against generalized linear attack, the paper is useful to study an application to see how the attack works.

The attack is also extended to a new attack called partitioning cryptanalysis. It is a more powerful attack which finds the last (or the first) round subkey. For the attack, the plaintext set and ciphertext set split into ”*partitions*”. ”The attack exploits a weakness that can be described by an effective partition-pair. That is, a pair of partitions such that, for every key, the next-to-last-round outputs are substantially non-uniformly distributed over the blocks of the second partition when the plaintexts are chosen uniformly from a particular block of the first partition.” [60]. It has been shown that partitioning cryptanalysis is more effective than linear cryptanalysis against variants of DES and CRYPTON. Useful papers to study on this subject are [63], [60] and [64].

4.3.2 Linear Cryptanalysis Using Multiple Linear Approximations

This extension of linear cryptanalysis is based on using more than one linear approximation at the same time. The idea is first proposed by Kaliski and Robshaw [58]. In the paper, the technique is briefly explained and an attack is mounted on DES. Although, the attack is advantageous since it uses less number of plaintext-ciphertext pairs than the usual linear cryptanalysis does, it is limited to the cases where all approximations derive the same parity bit of the key. This kind of problems of the attack is tried to be solved in [61].

In this section, we will just give the basic idea of linear cryptanalysis using multiple linear approximations [58]. Assume, n linear approximations are given which have the same parity bits of the round keys but different parity bits of the plaintexts and ciphertexts. The key bits that are in the approximations are determined by the following algorithm.

Algorithm 1M [58]

Let the i^{th} linear approximation for $1 \leq i \leq n$ be $P[l_1^i, l_2^i, ..., l_u^i] \oplus C[j_1^i, j_2^i, ..., j_v^i] = K[k_1, k_2, ..., k_w]$. Assume that the bias of each approximation ϵ_i is positive.

Step 1: Let T_i be the number of plaintext-ciphertext pairs such that the left side of the approximation is equal to 0. Let N denote the total number of plaintexts.

Step 2: For some set of weights $a_1, ...a_n$ where $\sum a_i = 1$ calculate $U = \sum a_i T_i$.

Step 3: If $U > N/2$ then guess $K[k_1, k_2, ..., k_w] = 0$, else guess $K[k_1, k_2, ..., k_w] = 1$.

It is shown that the new statistic U has a reduced variance compared to the T_i's. That is the reason that the attack requires fewer known plaintexts. Realize that the *Algorithm 1M* is an extension of Matsui's Algorithm 1 [55].

Also, an extension to Algorithm 1M [55] is given:

Algorithm 2M

Assume that n linear approximations are given for an r-round Feistel cipher such that we approximate from the second round to the $(r-1)^{th}$ round. The aim is to obtain some bits of the first round subkey and the last round subkey. All the n approximations should involve the same parity of subkey bits in the first and in the r^{th} round. Also, each of them is assumed to have a positive bias ϵ_i and of the form: $P[l_1, ..., l_u] \oplus C[j_1, ..., j_v] \oplus F_1(P_R, K_1)[m_1, ..., m_y] \oplus F_r(C_R, K_r)[s_1, ..., s_z] = K[k_1, ..., k_w]$.

Step 1: Let K_1^g $(g = 1, 2, ...)$ and K_r^h $(h = 1, 2, ...)$ be possible candidates for K_1 and K_r respectively. Then, for each pair (K_1^g, K_r^h) and each linear approximation i let $T_{g,h}^i$ be the number of plaintexts such that the left side of the approximation is equal to 0 when K_1 is replaced by K_1^g and K_r by K_r^h. Let N be the total number of plaintexts.

Step 2: Let ϵ_i be the bias of the i^{th} approximation. Let $a_i = \epsilon_i / \sum \epsilon_i$. Calculate for each g and h,

$$U_{g,h} = \sum_{n}^{i=1} a_i T_{g,h}^i$$

Step 3: Let U_{max} be the maximum value and U_{min} be the minimum value of all $U_{g,h}$.

- If $|U_{max} - N/2| > |U_{min} - N/2|$, adopt the key candidate corresponding to U_{max} and guess $K[k_1, ..., k_w] = 0$.

- If $|U_{max} - N/2| < |U_{min} - N/2|$, adopt the key candidate corresponding to U_{min} and guess $K[k_1, ..., k_w] = 1$.

Algorithm 1M and Algorithm 2M are both confirmed by a simple experiment on DES and it is observed that the success of a linear attack rises when two linear approximations are used instead of one. It is concluded that the attack offers an improvement in the number of plaintext-ciphertext pairs required for the linear cryptanalysis of a block cipher. However, the attack [58] on DES is somehow limited.

CHAPTER 5

MEASURES OF SECURITY AGAINST DIFFERENTIAL CRYPTANALYSIS, LINEAR CRYPTANALYSIS AND THEIR VARIANTS FOR BLOCK CIPHERS

In this section, we will discuss practical and provable security against differential cryptanalysis, linear cryptanalysis and their variants. We divided this section into parts. In the first part, the security of block ciphers against differential and linear cryptanalysis will be investigated. Then, in the remaining parts the resistance against truncated differential cryptanalysis, higher order differential cryptanalysis and generalized linear cryptanalysis will be examined.

5.1 Security Against Differential and Linear Cryptanalysis

Differential and linear attacks are powerful, basic attacks. A cipher must be designed as secure against these attacks. The security measures can be estimated from the attacker's stand point or the designer's stand point. To illustrate, the attacker tries to find high probability characteristics, in order to estimate the success rate of the attack and the computational load. However, this cannot be an approach for a designer to estimate the security for a cipher. Instead, designers try to show that the upper bound of the probability is sufficiently low for proving that the cipher is secure.

There exists four measures in order to evaluate the security of a cipher against differential and linear attacks. The designer can prove that his cipher is sufficiently invulnerable against these attacks by considering these four measures. Unfortunately, these evaluations do not prove that the cipher is secure against other known attacks. For instance, the KN cipher is proven to be secure against differential and linear cryptanalysis by using one of the four measures (theoretical measure). However, as it is told previously that, it can be broken by higher order differential attack.

- **The Precise Measure:** The maximum average of differential and linear probabilities. They are also called differential probability [33] and approximate linear hull [34]. Although in [33] and [34] it is stated that the precise evaluation of the security of a given block cipher should be done using this measure, it should be remarked that the computations of these probabilities are impractical.

- **Theoretical Measure:** The upper bounds of the maximum average of differential and linear probabilities. Feistel ciphers evaluated with this measure are theoretically invulnerable to differential and linear cryptanalysis [23]. This measure can be applicable to block ciphers. To illustrate, security of MISTY [35] and the KN cipher [23] are evaluated by applying this measure.

- **Heuristic Measure:** The maximum differential and linear characteristic probabilities. The security of a block cipher is characterized by the differential and linear characteristics. If a given block cipher is resistant to differential cryptanalysis and linear cryptanalysis, then the differential and linear characteristics of it have very small probabilities. So, the designers should intend to construct round functions yielding extremely small values of the probabilities. However, this is a very strong constraint on the design.
  Heuristic measure is applicable in practice. For instance, it is used to estimate the security of DES [36] and FEAL [37, 38]. We should remark that, the estimation for the probabilities takes much time, and for some ciphers, these probabilities only give the lower bounds of the maximum average of differential and linear probabilities [31].

- **Practical Measure:** The upper bounds of the maximum differential and linear characteristic probabilities. Feistel ciphers evaluated with this measure are practically secure against differential cryptanalysis and linear cryptanalysis [39].

Previously, we remarked that the maximum differential probability of a cipher is difficult to compute. Then, this question rises: Can we estimate the security of a cipher, by investigating the security of its round function? Actually, the answer is positive for DES-like block ciphers. In [23], it is proven that the maximum differential probability, denoted by DP_{max} of Feistel ciphers with four rounds can be estimated from the maximum differential probability of one round, denoted by dp_{max} as $DP_{max} \leq 2dp_{max}^2$. Also, it is shown in [40] that, if the round function is a permutation, the relation $DP_{max} \leq dp_{max}^2$ holds for Feistel ciphers with three rounds. There has been some studies for constructing practically secure round functions that yield sufficiently small of the maximum differential probabilities. Two illustrations for these can be found in [31] and [32].

There are several ways to determine the security of a block cipher against these two powerful attacks. To illustrate, in [43, 44] the notion of 'diffusion order' which enables to get lower bound on the number of active S-boxes in a substitution-permutation network is introduced. This is important since it is well known that the more the number of active S-boxes, the smaller the characteristic probability. Also, the theorems in [23] and [45] introduce ways to prove the security. Moreover, 'decorrelation theory' is an important notion to create provably secure ciphers against these atacks [46, 47, 48, 49, 50].

5.2 Security Against the Variants of Differential and Linear Cryptanalysis

5.2.1 Security Against Truncated Differential Cryptanalysis

The security against truncated differential cryptanalysis can provide a more strict evaluation of the security against differential cryptanalysis because for a truncated differential, the attacker only needs to predict parts of the bits of differences instead of all bits. To say that a cipher offers a security against this attack, one should search for the truncated differentials that can be used to mount an attack. This can be done by constructing search algorithms. By this method, the designer cannot always be sure that his cipher is secure against truncated differential attack. An illustration for this kind of estimation of security can be found in [41]. In this paper, the security of E2 against truncated differential cryptanalysis is studied. Firstly, an algorithm computing the probabilities for all truncated differentials of the round function with the SPN structure is described. Then, another algorithm which searches for all truncated differentials that lead to possible attacks of Feistel ciphers is introduced. By these algorithms, truncated differentials of E2 are searched and then, using the high probability ones among these differentials, possible scenarios of the attacks are described and the required complexities for these attacks are estimated. At the end, it is concluded that E2 is secure against truncated differential attack because by the algoritms, full round E2 is failed to be break. Moreover, a similar algorithm to the one in [41], is used to prove the security of Camellia against truncated differential cryptanalysis [27]. This time it is tried to find the upper bound of best bytewise characteristics of Camellia. Using this method no effective truncated characteristics more than seven rounds can be found. However, in [78], a nontrivial 9-round truncated differential that leads to a possible attack of reduced-round version of Camellia is introduced. This shows the algorithms described in the previous studies are not so effectice for Camellia. So, the designers should be careful while definning and applying search algorithms of truncated differentials for their ciphers.

5.2.2 Security Against Higher Order Differential Cryptanalysis

Block ciphers with provable security against differential cryptanalysis and linear cryptanalysis do not guarantee their security against higher order differential cryptanalysis. That is, this attack provides a security evaluation aspect different from that of provable security against differential cryptanalysis and linear cryptanalysis.

The success of higher order differential attack depends on the nonlinear order of S-box outputs. So, security against this attack can be evaluated by computing the degree of output bits after some rounds by considering the nonlinear order of the S-boxes. For instance, the polynomial degree of output bits after 3 rounds is found as 6^3 for CRYPTON, since the nonlinear order of the S-boxes are 6. So, one can say that CRYPTON is secure against higher order differential cryptanalysis after four rounds by using this observation.

5.2.3 Security Against Generalized Linear Cryptanalysis

Generalized linear cryptanalysis is known to be a more powerful attack than the ordinary linear cryptanalysis in some cases. So, proving a cipher's security against linear cryptanalysis is not enough to prove the resistance of it against generalized linear cryptanalysis.
Estimating a cipher's security against generalized linear cryptanalysis, one should determine a low upper bound for the avarage key imbalance of the I/O sums of the cipher [63].
An expression that serves an overall measure for the security of a cipher against partitioning cryptanalysis, ordinary and generalized linear cryptanalysis is presented. However, it is infeasible to compute the result of this expression if the input value is greater than or equal

to 64-bits. So, it is still an open problem to find a practically computable upper bound for today's ciphers [64].

To built a secure cipher against generalized linear cryptanalysis, it is suggested to use bent functions in the round function due to their low valued Fourier power spectra. Also, it is noted that the best candidates for the effective I/O sums are the homomorphic I/O sums. So, the designer should be careful about this [63].

The security of a cipher against linear cryptanalysis using multiple approximations has not been studied, yet. It is suggested that the Fourier approach presented in [63] and [64] can be generalized to cover this subject.

CHAPTER 6

ALGEBRAIC ATTACK

Algebraic attack is a cryptanalysis method that is efficient on block ciphers, stream ciphers and hash functions. It is significant for requiring only a handful of known plaintexts to perform. Actually, in the previous chapters we have seen that one needs so many plaintext-ciphertext pairs to mount a differential attack or a linear attack and that makes the cryptanalysis irrelevant in a realistic setting. That is, no adversary can possibly have that much pairs for real. However, sometimes algebraic cryptanalysis can be succesfully mounted even with only one plaintext-ciphertext pair. Actually, differential and linear attacks are faster than algebraic attacks generally but they require far more number of known plaintext-ciphertext pairs. Algebraic cryptanalysis promises successful attacks with a very low number of pairs.

The algebraic attack consists of two parts, obtaining a representation of the cipher as a system of equations and then, solving this system. According to Shannon, the effort that is spent for breaking a cipher is equivalent to the effort that is spent for solving a system of simultaneous equations in a large number of unknowns. So, if the attacker obtains a representation of the cipher as a system of equations and solves this, then a key recovery or retrieve of the plaintext can be possible. Although, in theory, most of the block ciphers can be fully expressed by a system of multivariate polynomials over a finite field, in practice this is a very complex study for most of the block ciphers. It is efficient to express each component of the round function as a polynomial, then find an expression for the whole cipher by connecting them. The attacker should try to find as much linearly independent equations as possible for the non-linear components. Actually, converting a cipher into a system of multivariate polynomial equations can be done in many different ways depending on how the attacker will mount the attack. There is no general method to do this so, we will just give an illustration on Keeloq .

We will focus only on the attacks on block ciphers and everything that we write are all about algebraic attacks on block ciphers. In the first section, we will give some methods to solve the system of multivariate polynomial equations. In the second section, we will report some important attacks and give an illustration of converting a cipher into a system of multivariate polynomial equations.

In this chapter, we will use the following notations:

r: Number of equations.

s: Number of unknowns in a system of equations.

t: Number of monomials in a system of equations (constant term is counted).

d: Specific degree of equations in a system.

Before beginning the sections the following definitions can be useful for the reader.

Definition 6.0.1 *A system of equations is called overdefined, if the number of equations in the system is greater than the number of unknowns. Namely, when $r >> s$.*

Definition 6.0.2 *A system of equations is called sparse, if $t << \binom{s}{d}$.*

It is advantageous for an attacker if the system is sparse or overdefined because these systems are easier to solve and actually, there exists techniques to solve these kind of systems which we will mention in the next section.

6.1 Methods for Solving the Polynomial Systems

For algebraic cryptanalysis, first the whole cipher is tried to be expressed by a system of low-degree multivariate polynomial equations, if possible. These systems involve a known plaintext-ciphertext pair (usually just one pair is enough for an attack), the secret key and intermediate variables appearing from the round operations. The attacker can obtain the secret key if he solves the system.

To write the cryptanalysis of a cipher as a problem of solving a system of multivariate quadratic equations is called MQ problem. If the equations of the system has degree at least two, then the problem is called MP problem. MQ and MP problems are both NP-Hard. Due to this fact, actually no polynomial time algorithms exist to solve these kind of systems. However, there exists some methods that are efficient to solve the systems in subexponential running time.

6.1.1 Linearization

Linearization is a principle that creates key stones for some algorithms such as XL and XSL algorithms. It is generally used in cryptanalysis of some LFSR-based stream ciphers. Linearization is a weak method to mount an attack on a modern block cipher but it is useful to study this in order to understand the other methods that depends on linearization.

Assume, we have given a system of multivariate quadratic equations. To solve this system with linearization, the attacker considers each of the degree two monomial as a new variable, gives a name to them and writes the new system. That is, a new system is constructed by replacing each degree two monomial, $x_i x_j$, by an independent degree one monomial, y_{ij}. All the equations in this new system are linear and the system can be solvable using Gaussian Elimination. It is important to note that the solutions to the linear system of equations (new system) may not be a solution to the original polynomial system. This can be understandable in a better way with an example.

Example:

$$x_1 + x_1x_2 + x_2 = 1$$

$$x_1 + x_1x_2 + x_3 = 0$$

$$x_1x_2 + x_2 + x_3 = 1$$

$$x_1x_2 + x_3 = 1$$

$$x_1 + x_2 + x_3 = 0$$

Assume we have given this simple system of multivariate quadratic equations. We will solve this system using linearization method. First, we will apply these substitutions on the system: $x_1 = x$, $x_1x_2 = y$, $x_2 = z$ and $x_3 = t$ to get the following new system:

$$x + y + z = 1$$

$$x + y + t = 0$$

$$y + z + t = 1$$

$$y + t = 1$$

$$x + z + t = 0$$

This system is linear and equivalent to the matrix system:

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{t} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

which can be reduced to the system:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{t} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

So, the solutions to this new system is $x = 1$, $y = 0$, $z = 0$, and $t = 1$. From here, we get the results, $x_1 = 1$, $x_2 = 0$ and $x_3 = 1$ which is a solution to the original system.

This example is actually very simple and troubleless. One can come face to face with some problems while solving any other systems. To illustrate, it is significant to note that any solutions to the new linear system of equations may not be a solution to the original polynomial system. On the contrary, a solution to the original system is always a solution to the new one. If a solution to the original polynomial system of equations exists, this solution can be obtained by trying all the solutions of the new system one by one. Also, one should note that, there is a case where no solutions exists for the linear system.

One should understand that by linearization the degree of a system of any degree can be decreased to two using the following step repeatedly [74]:

$$\{k = xyzw\} \rightarrow \{p = xy, q = zw, k = pq\}$$

6.1.2 The Extended Linearization (XL) Technique

XL is based on linearization and it is efficient to solve overdefined multivariate systems. It is firstly introduced in [67]. XL aims to expand the system by multiplying each polynomial equation with all the possible monomials of some bounded degree and then, solve it by linearization. The time complexity is claimed to be polynomial when the number of equations is at least square of the number of varibles multiplied with some constant, that is when $r \approx cs^2$ where c is a constant value [67]. However, the direct application of the method is known to be inefficient in practice for the recent block ciphers.

The algorithm is designed to solve a system of quadratic equations which have a solution in k^n for a finite field k. Let A be a system of multivariate equations $h_i = 0$ $(1 \leq i \leq r)$ for $h_i \in k[x] = k[x_1, x_2, ..., x_s]$. Let I_A be the ideal that is generated by all $h_i \in A$.

The XL Algorithm [67]

Let D be a positive fixed integer. Perform the following steps:

1. Multiply: Generate all the products
$$\sum_{j=1}^{k} x_{i_j} * h_i \in I_A,$$
where $k \leq D - 2$.

2. Linearize: Consider each monomial x_i that have degree smaller than or equal to D as a new variable and execute Gaussian Elimination on the equations that are obtained in 1. The ordering on the monomials must be adjusted such that all the terms containing one variable are eliminated last.

3. Solve: Solve the univariate equation that is obtained in step 2 over the finite fields by Berlekamp's algorithm.

4. Repeat: Simplify the equations and repeat the process to find the values of the other variables.

To understand the algorithm better, the following simple example may be helpful.

Example:

We will solve the following system of equations using XL method. The system is the same with the one that is given in the example of the linearization.

$$x + xy + y = 1$$

$$x + xy + z = 0$$

$$xy + y + z = 1$$

$$xy + z = 1$$

$$x + y + z = 0$$

We choose $D = 3$. The highest degree of the equations in the system is 2, so we have no other choise anyway. Now, multiply the equations in the system to increase their degree at most 3 to get these equations:

$$x + xy + y = 1$$

$$x + xy + xy = x$$

$$xy + xy + y = y$$

$$xz + xyz + yz = z$$

$$x + xy + z = 0$$

$$x + xy + xz = 0$$

$$xy + xy + yz = 0$$

$$xz + xyz + z = 0$$

$$xy + y + z = 1$$

$$xy + xy + xz = x$$

$$xy + y + yz = y$$

$$xyz + yz + z = z$$

$$xy + z = 1$$

$$xy + xz + x = x$$

$$xy + yz + y = y$$

$$xyz + z = z$$

$$x + y + z = 0$$

$$x + xy + xz = 0$$

$$xy + y + zy = 0$$

$$xz + yz + z = 0$$

$$xy + xy + xyz = 0$$

$$xz + xyz + xz = 0$$

$$xyz + yz + yz = 0$$

After simplifying these equations, we write the equivalent system with matrices:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{xy} \\ \mathrm{xz} \\ \mathrm{yz} \\ \mathrm{xyz} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Note that, we have considered each of the variables *xy*, *xz*, *yz* and *xyz* as new variables. That is, we have perfomed linearization on this variables secretly. This system is equivalent to the following system:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{xy} \\ \mathrm{xz} \\ \mathrm{yz} \\ \mathrm{xyz} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This system obviously yields the solution, $x = 1$, $y = 0$ and $z = 1$.

6.1.3 The Extended Sparse Linearization (XSL) Technique

The XSL algortihm is a variant of XL method and introduced in [66]. In the XL algorithm, the multivariate quadratic equations are multiplied by all the possible monomials with degree smaller than $D - 2$ but in XSL algorithm the equations are multiplied by only wisely selected monomials. Also, in contrast to XL method, the XSL technique aims to solve the system uniquely. On addition, XSL attack exploits the sparse equations in the system.

When XSL is first presented, it attracted a lot of attention because it was claimed that succesful attacks on AES could be mounted using XSL. However, studies has shown that the algorithm does not reduce the effort to break AES in comparison to an exhaustive search. Although the XSL attack does not work as expected, it is observed that AES could be expressed as a system of quadratic equations [66] and some cryptographers have expressed unease at the algebraic simplicity of ciphers like AES.

Assume, Ω is the system expressing the cipher that the attacker will mount an attack. In XSL method, the attacker splits the system Ω into small systems, say Ω_1,Ω_2,...Ω_n. Then, multiplies all the equations in Ω_i by the terms appearing in the other systems, Ω_j such that $j \neq i$. After applying a final step for increasing the number of linearly independent equations, linearization method is applied to solve the system. For the detailed explanation and algorithm of the method, please refer to [66]. Now, we will give a very simple example for this method.

Example:

We will solve the same example as the previous ones with XSL method. We first divide the system of equations into two parts. We will name the first group of equations as Q_1 and the other Q_2:

$$\mathrm{Q_1}=\begin{cases} \mathrm{x+xy+y=1} \\ \mathrm{x+xy+z=0} \\ \mathrm{x+y+z=0} \end{cases}$$

$$Q_2=\begin{cases} \mathrm{xy+y+z=1} \\ \mathrm{xy+z=1} \end{cases}$$

The set of the terms of the equations in system Q_1 is $P_1 = \{x, y, z, xy\}$ and the set of the terms of the equations in system Q_2 is $P_2 = \{xy, y, z\}$. Now, we will construct new equations by multiplying each element in P_1 by each equation in Q_2 and multiplying each element in P_2 by each equation in Q_1.

$Q_1 \cdot P_2$

$xy + xy + xy = xy$

$xy + xy + xyz = 0$

$xy + xy + xyz = 0$

$xy + xy + y = y$

$xy + xy + yz = 0$

$xy + y + yz = 0$

$xz + xyz + yz = z$

$xz + xyz + z = 0$

$xz + yz + z = 0.$

$Q_2 \cdot P_1$

$xy + xy + xz = x$

$xy + xz + x = x$

$xy + y + yz = y$

$xy + yz = y$

$xyz + yz + z = z$

$xyz + z = z$

$xy + xy + xyz = xy$

$xyz + xy = xy$

As one notices, there are linearly dependent equations in these systems. After, making some simplifications on the linearly independent equations we have the following matrix system:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{xy} \\ \mathrm{xz} \\ \mathrm{yz} \\ \mathrm{xyz} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

This system yields:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ \mathrm{xy} \\ \mathrm{xz} \\ \mathrm{yz} \\ \mathrm{xyz} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

From this system it is easy to see that the solution to the original system is $x = 1$, $y = 0$ and $z = 1$.

6.1.4 The ElimLin Technique

The ElimLin technique is firstly introduced in [68]. It can be seen as one of a simple version of a Grobner bases techniques. The ElimLin algorithm takes a system of degree two polynomial equations as an input and if the equations have a sufficient rank it gives a solution or solutions to this system. Also, if the equations do not have a sufficient rank, the algorithm outputs a reduced system of equations in fewer variables than the original and this system can be solved by some other method.

The ElimLin Algorithm

1. Linearize the input (system of quadratic equations).

2. Write the system as a matrix, then bring it into Row Reduced Echelon Form by using Gaussian Elimination.

3. If $r = 0$, stop. Else do:

- Elect to reduce the weight of the linear equations using some techniques (optional).
- For $i = 1...r$, process the following steps:
    (a) Define each variable by leaving it alone in the one side of the equation and moving all the other variables and constants to one side.
    (b) Eliminate one variable by substituting its definition that is obtained in the previous step into the other equations.
    (c) Store the definition of the variables, so that when the system is finally solved, the original ”eliminated” variables can be recovered.

4. Go back to step three and execute Gaussian Elimination.

This algorithm is repeated until no new linear equations can be obtained from the linear span of the equations in the system or until all the variables have been eliminated. Note that, one should first eliminate the variable that appears in the smallest number of equations to preserve sparsity [68].

6.1.5 SAT-Solvers

Converting MQ to CNF-SAT

SAT problem is defined as finding a satisfying assingnment for a logical expression in several variables. That is, SAT problem is the examination of the existence of a set of assignments of true and false to each of those variables so that the entire sentence evaluates as true when a logical sentence over certain variables is given and it is NP-hard. SAT problems can be solvable using SAT-solvers.

We know that all NP-complete problems are polynomially equivalent. So, considering that the MQ problem and the SAT problem are polynomially equivalent, the question is, can a sparse and overdefined system of polynomial equations be solved using SAT-solver tools. The answer is positive . In this section, we will investigate the usage of SAT-solvers in solving MQ problem which is proposed by Courtois et.al [74]. The fastest algebraic attacks are mounted using SAT-solvers [70][68]. The information in this section is a summary of the study that is described in [74]. For details, please refer to this paper.

For using SAT-solvers to solve the MQ problem, firstly the polynomial system of equations over $GF(2)$ should be converted into some other systems. This process consists of three steps:

1. Some preprocessing is performed before the conversion. By this part, the performance of the process is nearly doubled. Firsly, some monomials are redefined and these definitions are substituted instead of these monomials in the system. By this way, these monomials will not appear in any equation in the system except its definition, so that they are eliminated as variables. These monomials will be calculated lastly, since SAT-solvers starts to work on most frequently appearing variables.

2. The system is converted into a linear system and a set of conjunctive normal form (CNF) clauses that render each monomial equivalent to a variable in that linear system. Here, conjunctive normal form (CNF) of a logical sentence is a set of clauses. That is, it consists of variables and operators such as ”and” ($\wedge$), ”or” ($\vee$), ”implies” ($\rightarrow$), ”iff” ($\iff$) and ”not” ($\neg$). To illustrate, $\neg(A \vee B \vee \bar{C}) \rightarrow (C \wedge D)$ is a sentence in CNF. To make the system linear, for every monomial having degree more than one, a dummy variable is assigned to it and the definition is also added to the system as a new equation. Also, a variable is assigned to the constant values in the equation. Actually, in $GF(2)$ if a is a variable assigned for 1, then $\bar{a}$ is assigned for 0. By this, every equation has linear and higher degree variables and no constants. Then, these are converted to logical expressions. To illustrate, consider the equation $a = xyzw$ in $GF(2)$. This is tautologically equivalent to $a \Leftrightarrow x \wedge y \wedge z \wedge w$ and to $(x \vee \bar{a})(y \vee \bar{a})(z \vee \bar{a})(w \vee \bar{a})(a \vee \bar{x} \vee \bar{y} \vee \bar{z} \vee \bar{w})$.

3. The linear system is converted to an equivalent set of clauses. In this part, linear equations such as $x_1 \oplus x_2 \oplus ... \oplus x_q = 0$ are converted to clauses. To do this, firstly, the sum is divided into subsums of length four. That is, $x_1 \oplus x_2 \oplus ... \oplus x_q = 0$ can be divided as:

$$
\begin{gathered}
x_1 \oplus x_2 \oplus x_3 \oplus y_1 = 0 \\
y_1 \oplus x_6 \oplus x_7 \oplus y_2 = 0 \\
. \\
. \\
y_i \oplus x_{4i+2} \oplus x_{4i+3} \oplus y_{i+1} = 0 \\
. \\
. \\
y_f \oplus x_{q-2} \oplus x_{q-1} \oplus x_q = 0
\end{gathered}
$$

if q is an even number. Actually, if q is an odd number, then the last equation in the above system will be of length 3 and it is easier to convert a smaller length equation. These subsums of length 4 can be easily converted. For instance, the sum $x+y+z+w = 0$ is equivalent to;

$$(\bar{x} \vee y \vee z \vee w)(x \vee \bar{y} \vee z \vee w)(x \vee y \vee \bar{z} \vee w)(x \vee y \vee z \vee \bar{w})$$
$$(\bar{x} \vee \bar{y} \vee \bar{z} \vee w)(\bar{x} \vee \bar{y} \vee z \vee \bar{w})(\bar{x} \vee y \vee \bar{z} \vee \bar{w})(x \vee \bar{y} \vee \bar{z} \vee \bar{w})$$

One can calculate that sum of length q yields a certain number of subsums and each requires 8 clauses of length 4. Also, note that the subsums can be of length 3, 5 or higher according to desire. For this case, refer to [74].

An Illustration for SAT-Solvers: Walk-SAT

Walk-SAT is a randomized, easy to understand structured, simple SAT-solver proposed in [75]. It outputs a satisfying solution or runs forever. Before applying the algorithm, the system is converted into conjunctive normal form (CNF). Walk-SAT algorithm starts with an initial assignment. That is, it assigns a random value to each variable. The desired situation is that all the clauses are satisfied by the assignment. If all of them are satisfied then, the algorithm terminates. If not, then, a random unsatisfied clause is chosen and the random variable in that clause is flipped. This continues until all the clauses are satisfied. It is formally as follows:

Walk-SAT algorithm [76]

The algorithm takes a set of clauses, C and variables V, and three parameters p_{noise}, n_{flip} and $n_{restart}$ as input. It outputs either a satisfying solution or an abort.

1. For each variable in V, choose either 0 or 1 with equal probability.

2. For $i = 1, ..., n_{flip}$ do

   - If all clauses are satisfied, then output result and halt. Else, choose a violated clause $c \in C$, uniformly at random.

   - For each variable $v \in c$ do:
     −Compute the number of formerly satisfied clauses newly violated if v is flipped, denote this a_v.
     −Compute the number of formerly unsatisfied clauses newly satisfied if v is flipped, denote this b_v.
     −Substitute $a_v - b_v$ instead of k_v.

- With probability p_{noise} choose to flip a random variable in c.
- With probability $1 - p_{noise}$ choose to flip the v which minimizes the k_v.

3. If $n_{restart} = 0$ then, abort else decrement $n_{restart}$.
4. Restart the algorithm.

Also, there are other SAT-solvers such as MINI-SAT, G-SAT. MINI-SAT is more complex compared to Walk-SAT and it is based on logical principles. G-SAT is similar to Walk-SAT, they just differ in the methods used to select which variable to flip.

6.2 Some Illustrations of Algebraic Attack

Algebraic attack is a significant cryptanalysis method because it needs very low number of plaintext-ciphertext pairs which is a very striking feature for an attacker. Yet, most of the attacks on block ciphers are not feasible in practice. The recent methods for solving the multivariate quadratic systems should be developed or new methods should be proposed. In this section, we will mention two important and practically applicable illustrations of algebraic cryptanalysis. The first attack is mounted on DES [68] and the other is applied on KeeLoq [70].

6.2.1 An Algebraic Attack on DES

In this section, we will give a brief summary of the algebraic attack on DES reduced to six rounds [68]. DES is known to have a stronger algebraic structure as compared to AES but since the S-boxes of DES are small, they have a low I/O degree. The attacks published before the attack in [68] are weak as the description of the S-boxes are made using a ”functional” approach. This is the first ”convincing” algebraic attack on DES. The ElimLin method and SAT-solvers are used for the attack. It is shown that 6-round DES can be practically broken with only one plaintext-ciphertext pair.

For analysing the algebraic vulnerabilities of DES S-boxes, three certain classes of equations are investigated and it is seen that all the three classes give solvable systems of equations for varible rounds of DES. The number of linearly independent equations having variable degrees and types (fully cubic, sparse cubic, quadratic,...) for the DES S-boxes are computed.

Actually, these numbers are relevant for any 6×4 S-boxes. So, the attacks that are mounted using these equations are efficient for the other ciphers that are using modified DES S-boxes or any other 6×4 S-boxes.

For 4-round DES, a system consisting of 112 cubic equations per S-box is written with one plaintext. First 19 bits of the key are fixed to their real values. The correct solution to this system can be obtained in 8 seconds using ElimLin. Attacks on five rounds are claimed to be faster than brute force using again this method but we think it is not so significant.

For 6-round DES a system consisting of quadratic equations with additional variables are written using a simple ANF to CNF converter. Note that, 20 bits of the key are fixed. Then, the key is recovered in 68 seconds using MiniSat 2.0. The full 56 bits key can be obtained by this method with the complexity about 2^{48} applications of reduced DES which is feasible in practice.

6.2.2 Algebraic Attacks on KeeLoq

KeeLoq is a 32-bit block cipher with a very simple round function and 528 rounds. In each round, only one bit of the input state is modified. This may be the reason why the algebraic attack is succusfully applied on this block cipher. We can say that the attack in [70] is the most significant algebraic attack that can be practically mounted on a block cipher. In this section, first we will show how to obtain a representation of KeeLoq as a system of equations [73] and then, give a summary for the algebraic attack on KeeLoq [70].

Converting KeeLoq into System of Multivariate Polynomial Equations:

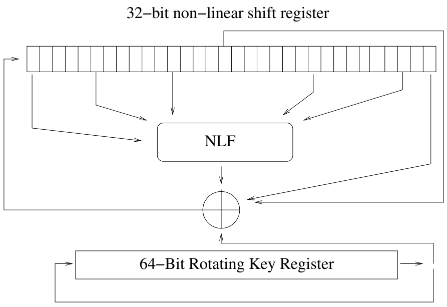

32−bit non−linear shift register
NLF
64−Bit Rotating Key Register

Figure 6.1: The Keeloq Circuit Diagram

The circuit diagram of KeeLoq is given 6.1. In the following steps there are some information about KeeLoq and some notation that is used for converting the cipher into system of equations.

- The 32-bit nonlinear shift register is initialized by a 32-bit plaintext block. In each round, the register shifts one to the right and one new bit comes to the left most position of the register. Stick to the paper [73], we will call the first 32 bit of the register as L_{31},L_{30},...,L_0. When register is shifted to the right, a new bit rises, and this bit is named as L_{32}. Similarly, the other new bits will be L_{33},L_{34}... and so on, lastly L_{559}. The ciphertext consists of the bits in the last attitude of the shift register. That is, $L_{559} = C_{31}$,...,$L_{528} = C_0$.
- The 64-bit rotating key register rotates only one key bit without modification in each round. Hence, in every 64 round the key bits will just be repeated. The 64 key bits are denoted as k_{63},k_{62},...,k_0. Then, the key bit corresponding to round i is actually $k_{i-1 mod 64}$.
- The non-linear function NLF takes (x, y, z, t, w) as an input where each of x, y, z, t and w are one bit values. If (x, y, z, t, w) is seen as a 5 bit integer, i, then the output is the value that is the i^{th} bit of the 32-bit hexadecimal value 3A5C742E. In [73], the algebraic normal form of NLF is obtained using Karnaugh map. Besides, the algebraic normal form can be directly obtained using the butterfly algorithm. It is the following:

$$NLF(x, y, z, t, w) = t \oplus w \oplus xz \oplus xw \oplus yz \oplus yw \oplus zt \oplus tw \oplus xtw \oplus xzw \oplus xyt \oplus xyz.$$

Following the above information, it is easy to write these equations:

- $L_i = P_i$, where $0 \leq i \leq 31$.
- $L_i = k_{i-32 mod 64} \oplus L_{i-32} \oplus L_{i-16} \oplus NLF(L_{i-1}, L_{i-6}, L_{i-12}, Li - 23, L_{i-30})$, where $32 \leq i \leq 559$.
- $C_i = L_{i-528}$ where $528 \leq i \leq 559$.

For each i there is an equation, so we have 560 equations for one plaintext-ciphertext pair. However, substituting some of the equations in the form of $L_i = P_i$ or $C_i = L_{i-528}$, the number of equations will drop down to 528. Also, there are 560 unknown variables, namely 64 unknown key bits and 496 unknown L_i values (32 last and 32 first of the 560 values of L_i are known). If the number of plaintext-ciphertext pairs increases, then the number of equations and the number of variables will also increase. For instance, if the number of the pairs is q, then there are $528q$ equations and $496q + 64$ variables since the key values remains the same while the pairs are changing.

Since the degree of the algebraic normal form of NLF is three, this system is a cubic system of equations. Using linearization, one can easily obtain a quadratic system of equations. For this purpose, let $\alpha = xy$ and $\beta = xw$ and determine

$$NLF(x, y, z, t, w) = t \oplus w \oplus xz \oplus \beta \oplus yz \oplus yw \oplus zt \oplus tw \oplus \beta t \oplus \beta z \oplus \alpha t \oplus \alpha z.$$

So, the system of equations is:

- $L_i = P_i$, where $0 \leq i \leq 31$.
- $L_i = k_{i-32mod64} \oplus L_{i-32} \oplus L_{i-16} \oplus L_{i-23} \oplus L_{i-30} \oplus L_{i-1}L_{i-12} \oplus \beta_i \oplus L_{i-6}L_{i-12} \oplus L_{i-6}L_{i-30} \oplus L_{i-12}L_{i-23} \oplus L_{i-23}L_{i-30} \oplus \beta_i L_{i-23} \oplus \beta_i L_{i-12} \oplus \alpha_i L_{i-23} \oplus \alpha_i L_{i-12}$, where $32 \leq i \leq 559$.
- $\alpha_i = L_{i-1}L_{i-6}$, where $32 \leq i \leq 559$.
- $\beta_i = L_{i-1}L_{i-30}$ where $32 \leq i \leq 559$.
- $C_{i-528} = L_i$ where $528 \leq i \leq 559$.

With two plaintext ciphertext pairs, we have 3168 equations and the same number of unknown variables.

The Attack

Some analysis of reduced round KeeLoq is made for complexity. By this way, simple structure of KeeLoq is investigated and this has been shown to the reader. The system of equations written for KeeLoq is sparse and the equations have very low degree. This system is tried to be solved by several algorithms, Magma's implementation of F4 algorithm [71], Singular's slimgb() algoritm [72], ElimLin and SAT-solvers. For the full rounds KeeLoq these methods are slower than the exhaustive search but for reduced rounds they work successfully. At most 128 rounds can be broken using these algorithms. Here are some results for the attacks of 128 rounds:

- Using ElimLin method, 34 bits of the key can be obtained in 3 hours with 128 rounds and 128 plaintexts in the counter mode and 30 bits fixed.
- Using MiniSat 2.0., 34 bits of the key can be obtained in 2 hours with 128 rounds, 2 plaintexts in the counter mode and 30 bits fixed.

More succesful results are obtained by mounting a cryptanalysis method that combines slide attack and algebraic attack on KeeLoq. That is, full 528 rounds can be broken using the combination of two cryptanalysis methods with 2^{16} known plaintexts. This is the first cryptanalytic attack on KeeLoq that works in practice. However, we will not give details since the slide attack is not included in the scope of this thesis.

REFERENCES

[1] National Institute of Standards and Technologies. *Data Encryption Standard*. In Federal Information Processing Standards Publication, FIPS-46-3, 1976.

[2] E. Biham, A. Shamir. *Differential cryptanalysis of DES-like cryptosystems*. Technical report CS90-16, Weizmann Institute of Science CRYPTO'90 and Journal of Cryptology, Volume 4, No. 1, pp. 3-72, 1991.

[3] B. V. Rompay, L. R. Knudsen, V. Rijmen. *Differential Cryptanalysis of the ICE Encryption Algorithm*. Lecture Notes in Computer Science, Volume 1372, Fast Software Encryption, pp. 270-283, 1998.

[4] H. M. Heys. *A Tutorial on Linear and Differential Cryptanalysis*. Technical Report CORR 2001-17, Centre for Applied Cryptographic Research, Department of Combinatorics and Optimization, University of Waterloo, 2001. (Also appears in Cryptologia, vol. XXVI, no. 3, pp. 189-221, 2002.)

[5] E. Biham, A. Shamir. *Differential Cryptanalysis of the full 16-Round DES*. Lecture Notes in Computer Science, Advances in Cryptology, CRYPTO'92, Volume 740, pp. 487-496, 1993.

[6] E. Biham, A. Shamir. *Differential Cryptanalysis of Snefru, Khafre, REDOC-2, LOKI and Lucifer*. Lecture Notes in Computer Science, Volume 576, Advances in Cryptology-CRYPTO'91, pp.156-171, 1992.

[7] I. B. Aroya, E. Biham. *Differential Cryptanalysis of Lucifer*. Journal of Cryptology, Volume 9, pp. 21-34, 1996.

[8] H. Raddum. *Differential Cryptanalysis of IDEA-X/2*. Lecture Notes in Computer Science, Volume 2887, Fast Software Encryption, pp. 1-8, 2003.

[9] L. R. Knudsen. *Cryptanalysis of LOKI91*. Lecture Notes in Computer Science, Volume 718, Advances in Cryptology, AUSCRYPT '92, pp. 196-208, 1993.

[10] X. Lai, J. L. Massey. *Markov Ciphers and Differential Cryptanalysis*. Advances in Cryptology, Lecture Notes in Computer Science, Volume 547, EuroCrypt'91, pp. 17-38, 1992.

[11] E. Biham, A. Shamir. *Differential Cryptanalysis of Feal and N-Hash*. Lecture Notes in Computer Science, Advances in Cryptology, Volume 547, EUROCRYPT'91, pp. 1-16, 1991.

[12] H. Yanami, T. Shimoyama, O. Dunkelman. *Differential and Linear Cryptanalysis of a Reduced Round SC2000*. Lecture Notes in Computer Science, Volume 2365, Fast Software Encryption, pp. 639-642, 2002.

[13] H.Raddum, L.Knudsen. *A Differential Attack on Reduced Round SC2000*. Lecture Notes in Computer Science, Volume 2259, Selected Areas in Cryptography, pp. 190-198, 2001.

[14] X. Lai. *Higher Order Derivatives and Differential Cryptanalysis*. Communications and Cryptography, Kluwer Academic Publishers, pp. 227-233, 1994.

[15] M. Matsui. *Linear Cryptanalysis Method for DES Cipher*. Lecture Notes in Computer Science, Advances in Cryptology-Eurocrypt'93, Volume 765, pp. 386-397, 1994.

[16] L. R. Knudsen. *Truncated and Higher Order Differentials*. Lecture Notes in Computer Science, Volume 1008, Fast Software Encryption - Second International Workshop, pp. 196-211, 1995.

[17] T. Jakobsen and L. R. Knudsen. *The Interpolation Attack on Block Ciphers*. Lecture Notes in Computer Science, Volume 1267, Fast Software Encryption - Fourth International Workshop, pp.28-40, 1997.

[18] L. McBride*Q: A proposal for NESSIE v2.00*. Firts NESSIE Workshop, Belgium, 2000.

[19] E. Biham, V. Furman, M. Misztal and V. Rijmen. *Differential Cryptanalysis of Q*. Lecture Notes in Computer Science, Volume 2355, FSE 2001, pp.311-321, 2002.

[20] A.W. Machado. *The Nimbus Cipher: A Proposal for NESSIE*. NESSIE Proposal, September, 2000.

[21] V. Furman. *Differential Cryptanalysis of Nimbus*. Lecture Notes in Computer Science, Volume 2355, pp. 113-125, 2002.

[22] T. Shimoyama, S. Moriai, and T. Kaneko. *Improving the Higher Order Differential Attack and Cryptanalysis of the KN Cipher*. Lecture Note in Computer Science, Volume 1396, Information Security, pp.32-42, Computer Science, 1998.

[23] K. Nyberg and L. R. Knudsen. *Provable Security Against a Differential Attack*. Journal of Cryptology, Volume 8, Number 1, Springer Verlag, pp.27-37, 1995.

[24] K. Nyberg. *Linear Approximations of Block Ciphers*. Lecture Note in Computer Science, Volume 950, Advances in Cryptology-EUROCRYPT'94, pp.439-444, 1995.

[25] Lars Knudsen, Thomas Berson. *Truncated Differentials of SAFER*. Lecture Notes in Computer Science, Volume 1039, Fast Software Encryption, pp. 15-26, 1996.

[26] L.R. Knudsen. *Truncated and Higher Order Differentials*. Lecture Notes in Computer Science, Volume 1008, Fast Software Encryption, pp. 196-211, 1995.

[27] M. Kanda, T. Matsumoto. *Security of Camellia against Truncated Differential Cryptanalysis*. Lecture Notes in Computer Science, Volume 2355, Fast Software Encryption, pp. 119-137, 2002.

[28] M. E. Hellman, S. K. Langford. *Differential-Linear Cryptanalysis*. Lecture Notes in Computer Science, Volume 839, CRYPTO'94, pp. 26-39, 1994.

[29] J. L. Massey. *SAFER K-64*: One year later. In B. Preneel Editor, Volume 1008, Fast Software Encryption, pp. 196-211, Springer Verlag, 1995.

[30] C. Harpes, G. G. Kramer, J. L. Massey. *A Generalization of Linear Cryptanalysis and Applicability of Matsui's Piling Up Lemma*. Lecture Notes in Computer Science, Volume 921, Advances in Cryptology-Eurocrypt'95, pp. 24-38, 1995.

[31] M. Kanda, Y. Takashima, T. Matsumoto, K. Aoki, K. Ohta. *A Strategy for Constructing Fast Round Functions with Practical Security against Differential and Linear Cryptanalysis*. Lecture Notes in Computer Science, Volume 1556, Selected Areas in Cryptography, pp.264-279, 1999.

[32] Y. Kaneko, F. Sano, K Sakurai. *On Provable Security against Differential and Linear Cryptanalysis in Generalized Feistel Ciphers with Multiple Random Functions*. Proceedings of SAC'97, pp. 185-199, 1997.

[33] X. Lai, J. L. Massey, S. Murphy. *Markov Ciphers and Differential Cryptanalysis*. Advances in Cryptology - Eurocrypt'91, Volume 547, 1991.

[34] K. Nyberg. *Linear Approximation of Block Ciphers*. Lecture Notes in Computer Science, Volume 950, Advances in Cryptology-Eurocrypt'94, pp.439, 1995.

[35] M. Matsui. *New Block Encryption Algorithm MISTY*. Fast Software Encryption-Fourth International Workshop, Volume 1267, 1997.

[36] M. Matsui. *Linear Cryptanalysis Method for DES Cipher*. Advances in Cryptology-Eurocrypt'93, Volume 765, 1994.

[37] K.Aoki, K. Kobayashi, S. Moriai. *Best Differential Characteristic Search of FEAL*. Lecture Notes in Computer Science, Volume 1267, Fast Software Encryption, pp. 41-53, 1997.

[38] K. Ohta, S. Moriai, K. Aoki. *Improving the Search Algorithm for the Best Linear Expression*. Advances in Cryptology - Crypto'95, Volume 963, 1995.

[39] L. R. Knudsen. *Practically Secure Feistel Ciphers*. Lecture Notes in Computer Science, Volume 809, Fast Software Encryption, pp. 211-221, 1994.

[40] K. Aoki, K. Ohta. *Strict Evaluation of the Maximum Average of Differential Probability and the Maximum Average of Linear Probability*. IEICE Trans., Volume E80-A, Number 1, pp. 2-8, 1997.

[41] S. Moriai, M. Sugita, K. Aoki, M. Kanda. *Security of E2 against Truncated Differential Cryptanalysis*. Lecture Notes in Computer Science, Volume 1758, Selected Areas in Cryptography, pp. 106-117, 2000.

[42] M.Sugita, K.Kobara, and H.Imai. *Pseudorandomness and Maximum Average of Differential Probability of Block Ciphers with SPN-Structures like E2*. In Proceedings of the Second Advanced Encryption Standard Candidate Conference, pp. 200-214, 1999.

[43] H. M. Heys. *The Design of Substitution-Permutation Network Ciphers Resistant to Cryptanalysis*. Ph.D. Thesis of Queen's University, Kingston, Ontario, Canada, 1994.

[44] H. M. Heys, S. E. Tavares. *Substitution-Permutation Networks Resistant to Differential and Linear Cryptanalysis*. Journal of Cryptology, Volume 9, pp. 1-19, Computer Science, 1996.

[45] K. Nyberg, L. R. Knudsen. *Provable Security against a Differential Cryptanalysis*. Lectures Notes in Computer Science, Volume 740, CRYPTO'92, pp. 566-574, 1993.

[46] S. Vaudenay. *A cheap Paradigm for Block Cipher Security Strengthening*. Technical Report LIENS-97-3, 1997.

[47] S. Vaudenay. *Provable Security for Block Ciphers by Decorrelation*. Lectures Notes in Computer Science, Volume 1373, STACS 98, pp. 249-275, 1998.

[48] S. Vaudenay. *Feistel Ciphers with L2-Decorrelation*. Lecture Notes in Computer Science, Volume 1556, Selected Areas in Cryptography, pp. 631, 1999.

[49] S. Vaudenay. *The Decorrelation Technique Home-Page*. http://www.dmi.ens.fr/ vaudenay/decorrelation.html.

[50] S. Vaudenay. *Resistance Against General Iterated Attacks*. Lecture Notes in Computer Science, Volume 1592, Advances in Cryptology-EUROCRYPT99, pp. 255-271, 1999.

[51] M. Matsui, A. Yamagishi. *A New Method for Known Plaintext Attack of FEAL Cipher*. Lecture Notes in Computer Science, Volume 658, Advances in Cryptology, Proceedings of Eurocrypt'92, pp. 81-91, 1993.

[52] M. Matsui. *Linear Cryptanalysis Method for DES Cipher*. Lecture Notes in Computer Science, Volume 765, Eurocrypt'93, pp. 386-397, 1993.

[53] B.S. Kaliski, Y.L. Yin. *On Differential and Linear Cryptanalysis of RC5 Encryption Algorithm*. Lecture Notes in Computer Science, Volume 963, Crypto'95, pp. 171-184, 1995.

[54] A.A. Selçuk. *New Results in Linear Cryptanalysis of RC5*. Lecture Notes in Computer Science, Volume 1372, Fast Software Encryption, pp. 1-16, 1998.

[55] M. Matsui. *Linear Cryptanalysis Method for DES Cipher*. Lecture Notes in Computer Science, Advances in Cryptology-Eurocrypt'93, pp. 386-397, 1993.

[56] J. Borst, B. Preenel, J. Vandewalle. *Linear Cryptanalysis of RC5 and RC6*. Lecture Notes in Computer Science, Volume 1636, Fast Software Encryption, pp. 16-30, 1999.

[57] Eli Biham. *On Matsui's Linear Cryptanalysis*. Lecture Notes in Computer Science, Volume 950, Advances in Cryptology-EuroCrypt'94, pp. 341-355, 1995.

[58] B. Kaliski and M. Robshaw. *Linear Cryptanalysis Using Multiple Approximations*. In Y.G. Desmedt, Editor, Volume 839, Advances in Cryptology - Crypto'94, pp. 26-39, 1994.

[59] C. Harpes, G. Kramer, and J. Massey. *A Generalization of Linear Cryptanalysis and The Applicability of Matsui's Piling-up Lemma*. Lecture Notes in Computer Science, Volume 921, Advances in Cryptology - Eurocrypt'95, pp. 24-38, 1995.

[60] C. Harpes and J. Massey. *Partitioning Cryptanalysis*. Lecture Notes in Computer Science, Volume 1267, Fast Software Encryption, pp. 13-27, 1997.

[61] A.Biryukov, C. D. Canniere and M. Quiaquater. *On Multiple Linear Approximations*. Lecture Notes in Computer Science, Volume 3152, Advances in Cryptology-CRYPTO'2004, pp. 1-22, 2004.

[62] C. Harpes. *Generalization of Linear Cryptanalysis Applied to SAFER*. Lecture Notes in Computer Science, Volume 921, Advances in Cryptology-EUROCRYPT95, pp. 24-38, 1995.

[63] T. Jacobsen.*Security Against Generalized Linear Cryptanalysis and Partitioning Cryptanalysis*. Semester Project at Signal and Information Processing Laboratoryi Swiss Federal Institute of Technology Zurich, Zürich, 1995.

[64] T. Jakobsen, C. Harpes. *Bounds On Non-Uniformity Measures For Generalized Linear Cryptanalysis And Partitioning Cryptanalysis*. Pragocrypt96. Prague: Czech Technical University Publishing House, pp. 467-479, 1996.

[65] C. E. Shannon. *Communication Theory of Secrecy Systems* Bell System Technical Journal 28, Volume 28, pp. 656-715, 1949.

[66] N Courtois, J. Pieprzyk. *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*. Lecture Notes in Computer Science, Volume 2501, Asiacrypt 2002, pp. 267-287, 2002.

[67] N. Courtois, A. Shamir, J. Patarin, A. Klimov, *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations*. Lecture Notes in Computer Science, Volume 1807, Eurocrypt'2000, pp. 392-407, 2000.

[68] N. Courtois, G. Bard. *Algebraic Cyptanalysis of the Data Encryption Standard*. Lecture Notes in Computer Science, Volume 4887, pp. 152-169, 2007.

[69] N. Courtois. *The Security of Cryptographic Primitives Based on Multivariate Algebraic Problems: MQ, MinRank, IP, HFE*. Ph.D. thesis, Paris 6, 2001. Available at http://www.nicolascourtois.net/phd.pdf.

[70] N. Courtois, G. V. Bard, D. Wagner. *Algebraic and Slide Attacks on KeeLoq*. Lecture Notes in Computer Science, Volume 5086, Fast Software Encryption, pp. 97-115, 2008.

[71] N. Courtois. *How Fast can be Algebraic Attacks on Block Ciphers?*. E. Biham, H. Handschuh, S. Lucks, V. Rijmen. Online Proceedings of Dagstuhl Seminar 07021, Symmetric Cryptography, 2007. Avaliable at http://drops.dagstuhl.de/portals/index.php?semnr=07021, http://eprint.iacr.org/2006/168/ ISSN 1862 - 4405.

[72] E. K. Grossman, B. Tuckerman. *Analysis of a Feistel-like cipher weakened by having no rotating key*, IBM Thomas J. Watson Research Report RC 6375, 1977.

[73] G. Bard. *Algorithms for the Solution of Linear and Polynomial Systems of Equations over Finite Fields, with Applications to Cryptanalysis.* Ph.D. thesis, Department of Applied Mathematics and Scientific Computation, University of Maryland at College Park, 2007.

[74] G. Bard, N. Courtois, C. Jefferson. *Efficient methods for conversion and solution of sparse systems of low-degree multivariate polynomials over GF(2) via SAT-Solvers*. Cryptology ePrint Archive, Report 2007/024, 2006. Available at http://eprint.iacr.org/2007/024.pdf

[75] H. Kautz, B. Selman, D. McAllester. *Walksat in the 2004 SAT Competition*. International Conference on Theory and Applications of Satisfiability Testing, Vancouver, 2004.

[76] B. Selman, H. Kautz, B. Cohen. *Local search strategies for satisfiability testing*. In: D.S. Johnson, M.A. Trick Editor, Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge (DIMACS'93), Volume 26, AMS 1996.

[77] R. L. Rivest. *The RC5 Encryption Algorithm*. Lecture Notes in Computer Science, Volume 1008, Fast Software Encryption, pp. 86-96, 1995.

[78] M. Sugita, K. Kobara, H. Imai. *Security of Reduced Version of the Block Cipher Camellia against Truncated and Impossible Differential Cryptanalysis*. Lecture Notes in Computer Science, Volume 2248, Advances in Cryptology-ASIACRYPT 2001, pp. 193-207, 2001.

[79] A. Darbuka. *Related-Key Attacks on Block Ciphers*. M.Sc. Thesis, 2009. Avaliable at http://www.iam.metu.edu.tr/mscthesis/aslidarbukathesis.pdf.

[80] N. Öztop. *Combined Attacks on Block Ciphers*. M.Sc. Thesis, 2009.

[81] A. Biryukov. *Methods of Cyptanalysis*. Ph.D. Thesis, 1999.

[82] D.W. Davies and W.L. Price. *Security for Computer Networks*. John Wiley and Sons, 1989.

[83] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, ISBN 0-8493-8523-7, 1996. Available at http://www.cacr.math.uwaterloo.ca/hac/.

APPENDIX A

DESCRIPTION OF SAMPLE CIPHER

The sample cipher is a Feistel type block cipher which takes 16 input bits and gives 16 output bits. It is running five rounds and each round comprises a round function, f, which consists of three components, namely key mixing, substitution of bits, and permutation of bit positions. In each round, f function takes 8 bits as an input and gives 8 bits output.
The key scheduling process is not important for this paper, so we will not mention about it but just assume that the subkeys that are generated from the cipher's master key are independent and unrelated.

A.1 Components of the Round Function

The sample cipher is designed to be breakable simply because it is aimed to describe the basic attaks differential cryptanalysis and linear cryptanalysis by mounting on the sample cipher. The round function is similar to the one used in DES. The components are given as follows.

Key Mixing Layer: The input bits are bit-wise XORed with the key bits of the corresponding round in this first component of the f function. The cipher uses the same subkey which is 8 bits long in each round.

Substitution Layer: After the key mixing, 8 bit data become the input of the substitution boxes (S-boxes) which compose the second component of the f function. S-boxes of the sample cipher is defined as a mapping with 4 input bits and 4 output bits. Each round includes 2 identical S-boxes which are defined in the following table A.1.

Table A.1: Substitution-Box

INPUT	0	1	2	3	4	5	6	7	8	9	*A*	*B*	*C*	*D*	*E*	*F*
OUTPUT	*A*	3	8	5	*C*	0	2	*F*	6	*E*	1	4	9	7	*D*	*B*

Permutation Layer: The output bits of the S-boxes are permuted. In other words, the positions of bits of the output of S-boxes are changed according the rule described in table A.2.

Table A.2: Permutation

INPUT	1	2	3	4	5	6	7	8
OUTPUT	7	6	8	5	2	1	4	3