Diss. ETH No. 9752

On the Design and Security of Block Ciphers

A dissertation submitted to the

SWISS FEDERAL INSTITUTE OF TECHNOLOGY

ZÜRICH

for the degree of

Doctor of Technical Sciences

presented by

XUEJIA LAI

B.Sc. El.Eng., M.Sc. Math. Xidian University, Xian, China

born June 4, 1954

citizen of China

accepted on the recommendation of

Prof. Dr. J. L. Massey, referee

Prof. Dr. H. Bühlmann, co-referee

Zürich, 1992

Hartung-Gorre Verlag Konstanz

Acknowledgements

I am deeply grateful to Prof. J. L. Massey who guided this work not only as an outstanding adviser but also as an understanding friend. I am further grateful to Prof. H. Bühlmann for his valuable comments.

I would like to thank my colleagues at the Signal and Information Processing Laboratory for the enjoyable atmosphere at the institute.

I also want to express my thanks to my friends and colleagues at ETH, Jürg Ganz, Carlo Harpes, Alain Hiltgen, Richard De Moliner, Ueli Maurer, Thomas Mittelholzer, Marcel Rupf, Rainer Rueppel, Christian Waldvogel for their encouragement, interest and comments.

Many thanks go to the research group at the Integrated Systems Laboratory, ETH, Prof. W. Fichtner, H. Kaeslin, N. Felber, H. Bonnenberg, A. Curiger and M. Halberherr for their indispensable support.

Special thanks go to Ascom Hasler AG, Bern, to Ascom Tech AG, Solothurn, and to the Kommission zur Förderung der Wissenschaftlichen Forschung, Bern, without whose financial help this work would not have been possible. In particular, I would like to thank Mr. C. Siuda for initiating this project, Dr. E. Schwerdtel for preparing the patent application for the IDEA cipher, and Dr. D. Profos, Mr. T. Brüggemann, Mr. H. Bürk, Dr. J. Piveteau and Mr. B. Stuber for their support to my work.

Finally, I want to thank my wife, Shuning, for her patience and understanding.

Abstract

Secret-key block ciphers are the subject of this work. The design and security of block ciphers, together with their application in hashing techniques, are considered. In particular, iterated block ciphers that are based on iterating a weak round function several times are considered. Four basic constructions for the round function of an iterated cipher are studied.

The iterated block cipher IDEA is proposed. This cipher is based on the new design concept of mixing different group operations on 16-bit subblocks. Using operations on subblocks facilitates the software implementation of the cipher. The regular structure of the cipher facilitates hardware implementation. The interaction of the three chosen "incompatible" group operations provides the necessary "confusion", and the chosen cipher structure causes the required "diffusion".

The security of iterated ciphers against Biham and Shamir's differential cryptanalysis is discussed. Differential cryptanalysis is described in terms of an i-round "differential", which is defined as a couple (α, β) such that a pair of distinct plaintexts with difference α can result in a pair of i-th round outputs having difference β. It is shown that the maximum probability of such a differential can be used to determine a lower bound on the complexity of a differential cryptanalysis attack. The concept of "Markov ciphers" is introduced because of its significance in differential cryptanalysis. It is shown that the security of a Markov cipher against differential cryptanalysis is determined by the transition probability matrix created by the round function. A design principle for Markov ciphers is formulated, viz., that its transition matrix should be non-symmetric. Differential cryptanalysis of the IDEA cipher is performed partly by theoretical analysis of the relationship between the three chosen group operations and the properties of the MA-structure within the cipher, and partly by numerical experiments on "mini versions" of the cipher.

The results suggest that the IDEA cipher is secure against differential cryptanalysis attack after only four of its eight rounds.

The application of block ciphers in constructing hash functions is also considered. Five different attacks on hash functions obtained by iterating a hash round function are formulated and examined. Relations between the security of such an iterated hash function and the strength of its round function are derived. Schemes for constructing hash round functions by using block ciphers are discussed and new hashing schemes using the IDEA cipher are proposed. In particular, the problem of constructing $2m$-bit hash round functions from available m-bit block ciphers is considered and two new constructions are proposed. Four attacks on three known hash schemes are presented by applying a new principle for evaluating the security of a hash round function.

Zusammenfassung

Diese Arbeit handelt vom Entwurf und der Sicherheit von Blockchiffrierern, sowie von desen Anwendung in Hash-Verfahren. Insbesondere werden die iterative Blockchiffrierer betrachtet, die auf mehrmaliger Wiederholung einer "schwachen" Rundenfunktion basieren. Vier grundlegende Konstruktionen dieser Rundenfunktion werden untersucht.

Das Blockchiffrierverfahren IDEA wird vorgeschlagen, welches mit Hilfe eines neuen Konzepts entworfen wurde. Es handelt sich dabei um eine Vermischung von unterschiedlichen Gruppenoperationen, die auf 16-Bit Teilblöcken operieren. Die Verwendung von Operationen auf Teilblöcken erleichtert die Software-Implementation dieses Chiffrierers, und die regelmässige Struktur des Chiffrierers ermöglicht eine effiziente Hardware-Implementation. Die Wechselwirkung der drei gewählten "inkompatiblen" Gruppenoperationen liefert die notwendige "Confusion", und die gewählte Struktur des Chiffrierers erzeugt die notwendige "Diffusion".

Die Sicherheit von iterativen Blockchiffrierern gegenüber der Differential-Kryptanalyse, welche von Biham und Shamir stammt, wird untersucht. Dabei wird die Differential-Kryptanalyse mit Hilfe eines i-Runden-Differentials (α, β) beschrieben, welches so definiert ist, dass ein Paar von verschiedenen Klartexten mit Differenz α nach i Runden ein Paar von Chiffriertexten mit Differenz β erzeugen kann. Es wird gezeigt, dass die maximale Wahrscheinlichkeit eines solchen Differentials benutzt werden kann, um eine untere Schranke der Komplexität einer Differential-Kryptanalyse-Attacke zu bestimmen. Der Begriff "Markov-Chiffrierer" wird wegen seiner Bedeutung für die Differential-Kryptanalyse eingeführt. Es wird gezeigt, dass die Sicherheit eines Markov-Chiffrierers durch die von der Rundenfunktion erzeugte Matrix der Übergangswahrscheinlichkeiten bestimmt werden kann. Ein Entwurfsprinzip für Markov-Chiffrierer wird formuliert, nämlich, dass die Matrix der Über-

gangswahrscheinlichkeiten asymmetrisch sein soll. Die Differential-Kryptanalyse des IDEA-Chiffrierers wird einerseits theoretisch durchgeführt, indem die Beziehungen zwischen den drei Gruppenoperationen und die Eigenschaften der verwendeten MA-Struktur analysiert werden. Andererseits liegen auch numerische Untersuchungen mit "Mini-Versionen" des IDEA-Chiffrierers vor. Die Ergebnisse dieser Untersuchungen legen den Schluss nahe, dass der IDEA-Chiffrierer nach nur vier von seinen acht Runden gegenüber der Differential-Kryptanalyse sicher ist.

Weiter wird die Anwendung von Blockchiffrierer zur Konstruktion von Hash-Funktionen besprochen. Fünf verschiedene Attacken auf Hash-Funktionen, die auf Wiederholungen einer Hash-Rundenfunktion beruhen, werden formuliert und untersucht. Zusammenhänge zwischen der Sicherheit einer solchen iterativen Hash-Funktion und der Stärke ihrer Rundenfunktion werden hergeleitet. Konstruktionsmethoden für Hash-Rundenfunktionen, welche auf einem Blockchiffrierer basieren, werden betrachtet. Neue Hash-Verfahren, welche auf dem IDEA-Chiffrierer beruhen, werden ebenfalls vorgeschlagen. Insbesondere wird das Problem der Konstruktion von $2m$-Bit Hash-Rundenfunktionen aus verfügbaren m-Bit Blockchiffrierern behandelt, und es werden zwei neue Konstruktionen vorgeschlagen. Vier Attacken auf drei bekannte Hash-Funktionen werden vorgestellt, die auf einem neuen Prinzip der Sicherheitsevaluierung von Hash-Rundenfunktionen beruhen.

Contents

List of Figures

Chapter 1

Introduction

In this work, we consider the design of practical secret-key block ciphers, the security of such ciphers and their applications in constructing hash functions.

In Chapter 2, the unconditional security and computational security of a secret-key block cipher are discussed. It is pointed out that one should distinguish the data-complexity from the processing-complexity of a specified attack in evaluating the security of a cipher against the attack. General design principles for a practical block cipher are summarized in Section 2.3 according to security requirements and implementation constraints. In particular, we consider the design of iterated block ciphers that are based on iterating a weak round function several times. Four basic constructions of iterated ciphers are presented in Section 2.4 for which the resulting ciphers have the property that encryption and decryption can be computed by the same process except for the use of different subkeys computed from the secret key.

A complete description of a proposed new block cipher, the IDEA cipher, is presented in Chapter 3. The IDEA cipher is based on the design concept of mixing different group operations on 16-bit subblocks. Using operations on subblocks facilitates the software implementation of the cipher and the regular structure of the cipher facilitates hardware implementation. The interaction of the three chosen "incompatible" group operations provides the necessary "confusion", and the chosen cipher structure causes the required "diffusion".

The security of iterated ciphers against Biham and Shamir's differential cryptanalysis is discussed in Chapter 4. Differential cryptanalysis is a chosen plaintext attack to find the secret key of an iterated cipher by analyzing the effect of the difference of a pair of plaintexts on the difference of succeeding round outputs in an r-round cipher. We describe differential cryptanalysis in terms of an i-round "differential" instead of in terms of an i-round "characteristic" as was done by Biham and Shamir. An i-round differential is defined as a couple (α, β) such that a pair of

distinct plaintexts with difference α can result in a pair of i-th round outputs having difference β. It is shown that the maximum probability of such a differential can be used to determine a lower bound on the complexity of a differential cryptanalysis attack. The concept of "Markov ciphers" is introduced because of its significance in differential cryptanalysis and because of the fact that many known secret-key block ciphers are Markov. It is shown that the security of a Markov cipher against differential cryptanalysis is determined by the transition probability matrix created by the round function. In particular, a design principle for Markov ciphers is formulated, viz., that its transition matrix should be non-symmetric.

Differential cryptanalysis of the IDEA cipher is considered in Chapter 5. Based on the relationship between the three chosen group operations, the properties of the MA-structure within the cipher and numerical experiments on "mini versions" of the cipher, the security of the IDEA cipher against differential cryptanalysis is analyzed by determining the properties of the transition matrix of the cipher and by finding the plausibly most probable differentials. The analysis suggests that the IDEA cipher is secure against differential cryptanalysis attack after only four of its eight rounds.

The application of block ciphers in constructing hash functions is considered in Chapter 6. A rather rounded treatment of hash functions obtained by iterating a hash round function is given in Section 6.2, where we examine five different attacks on such iterated hash functions and consider relations between the security of an iterated hash function and the strength of its round function. It is shown that, by constraining the last block of the message to be hashed, secure hash function can be obtained from a secure round function. Schemes for constructing hash round functions by using a block cipher are considered. Three new hashing schemes based on the IDEA cipher are proposed. In particular, we consider the problem of constructing $2m$-bit hash round functions from available m-bit block ciphers. A principle is formalized for evaluating the strength of hash round functions, viz., that applying simple (in both directions) invertible transformations to the input and output of a hash round function will yield a new hash round function with the same security. This principle is demonstrated via four "meet-in-the-middle" attacks on three known $2m$-bit hash round functions based on an m-bit block cipher.

Chapter 2

Introduction to Block Ciphers

2.1 Secret-Key Block Ciphers

The vast amount of information transmitted over today's ever growing communication and computer networks demands protection from undesirable disclosure, i.e., *secrecy*, and protection from forgery and undesirable altering, i.e., *authenticity*. Cryptographic techniques are required to provide such secrecy and authenticity [36].

Cryptosystems can be divided into *secret-key* systems and *public-key* systems. In a secret-key system, the legitimate users (sender and receiver) must share a secret (the secret key) that is unknown to the enemy cryptanalyst. In a public-key system, the legitimate users need only some trusted information in common. Although public-key systems appear to be ideal for many cryptographic applications, their low speed and high cost bar their use in many situations. In this dissertation, only secret-key cipher systems will be discussed.

We will use Shannon's model of a secrecy system [53], which is shown in Fig.2.1. In this model, the secret key Z is distributed to the sender and receiver via the secure channel. This key is then used to encrypt the plaintext X into the ciphertext Y by the sender and to decrypt the ciphertext Y back to the original plaintext X by the receiver. The ciphertext is transmitted over an insecure channel for which it is assumed that the enemy cryptanalyst has the same access as does the legitimate receiver. The secret key, however, is inaccessible to the cryptanalyst. Such a secret-key cryptosystem is also called a *symmetric* cryptosystem to distinguish it from an *asymmetric* public-key cryptosystem in which different keys are used by the encrypter and by the decrypter. Note that X, Y and Z in this model are random variables. As Fig.2.1 suggests, we always assume that the plaintext X and key Z are statistically independent.

Secret-key cipher systems are usually divided into block ciphers and stream ci-

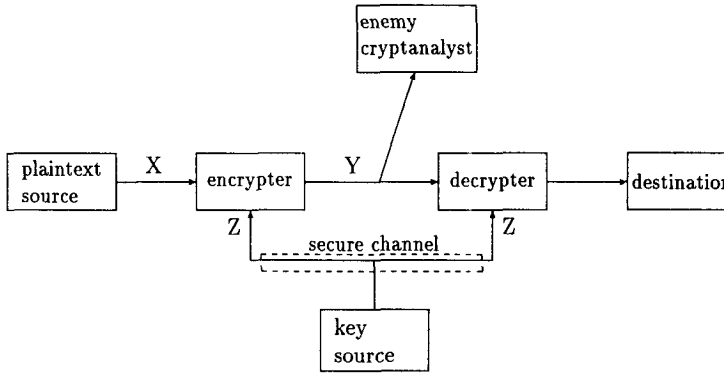


Figure 2.1: A cipher system–Shannon's model of secret communication.

phers. For a block cipher, the plaintext has the form of "large" (e.g., 64-bit) blocks and consecutive blocks are encrypted by the same encryption function, i.e., the encrypter is a memoryless device. In a stream cipher, the plaintext is usually a sequence of "small" (typically 1-bit) blocks that are transformed by an encrypter with memory.

Block ciphers have the advantage that they can be easily "standardized", as information is usually processed and transmitted in the form of "blocks" (e.g., bytes or words) in today's data networks. Ease of synchronization in the sense that losing one ciphertext block has no influence on the correctness of the decryption of the following blocks is another advantage of using block ciphers.

The main disadvantage of a block cipher is that block encryption does not "hide" data patterns: identical ciphertext blocks imply identical plaintext blocks. However, this disadvantage can be overcome by introducing a small amount of memory in the encryption process, e.g., by using the Cipher Block Chaining (CBC) mode [14] in which the memoryless encryption function is applied to the XOR-sum of the plaintext block and the previous ciphertext block. The resulting cipher is now technically a "stream cipher" that employs "large" blocks.

Let $\mathbb{F}_2$ denote the Galois field of two elements and let $\mathbb{F}_2^m$ denote the vector space of m tuples with elements in $\mathbb{F}_2$. In this work, we will consider in most cases only binary ciphers without real loss of generality, i.e., we assume that the plaintext X and the ciphertext Y take on values in the vector space $\mathbb{F}_2^m$ and that the key Z takes values in a subset S_z of the vector space $\mathbb{F}_2^k$. Thus, m is the length in bits of

the plaintext and ciphertext blocks, and k is the length in bits of the secret key.

Definition 1 *A secret-key block cipher is a mapping* $E : \mathbb{F}_2^m \times S_z \rightarrow \mathbb{F}_2^m$ *such that, for each* $z \in S_z$, $E(\cdot, z)$ *is an invertible mapping from* $\mathbb{F}_2^m$ *to* $\mathbb{F}_2^m$.

The invertible function $E(\cdot, z)$ will be called the *encryption function* for the key z. The inverse of $E(\cdot, z)$ will be called the *decryption function* for the key z and will be denoted as $D(\cdot, z)$. We will write $Y = E(X, Z)$ for a block cipher to mean that the ciphertext Y is determined by the plaintext X and secret key Z via the mapping E. The parameter m is the *block length* of the block cipher and the parameter k is the *key length* of the block cipher. The *true key size* of the block cipher is defined as $k_t = \log_2(\#(S_z))$ bits. Thus, the key length is the true key size if and only if $S_z = \mathbb{F}_2^k$, i.e., if and only if every binary k-tuple is a valid key. For instance, the block cipher DES [14] has a key length of $k = 64$ bits, but a true key size of only $k_t = 56$ bits.

Note that the block ciphers we will consider in this work are those without plaintext expansion, i.e., the plaintext block and the ciphertext block have the same length, which is virtually always a requirement in applications.

2.2 Security of Block Ciphers

As stated before, a cipher can be used to provide protection from undesirable disclosure of plaintext. The task of an enemy cryptanalyst, the attacker, is then to "break" the cipher in the sense that he can recover the plaintexts from intercepted ciphertexts. A cipher is totally broken if the cryptanalyst can determine the secret key in use so that he can read all messages thereafter as easily as the legitimate user can. A cipher is partially broken if the cryptanalyst can frequently recover the plaintexts from intercepted ciphertexts, but cannot find the secret key.

Security is always *relative* to threats. As stated above, we have assumed that the attacker has access to everything transmitted through the insecure channel. However, there may be other information available to the cryptanalyst. The computational capability of the cryptanalyst must also be considered before the security of a cipher can be assessed.

2.2.1 Types of attacks

The most universally accepted assumption in cryptography is that the attacker (enemy cryptanalyst) has full access to the ciphertext transmitted over the insecure channel. Another well-accepted assumption is the following:

Kerckhoff's assumption: The enemy cryptanalyst knows all details of the process of encryption and decryption except for the value of the secret key.

Kerckhoff's assumption implies that the security of a secret-key cipher system rests entirely on the secret key. Under Kerckhoff's assumption, attacks are usually classified according to the cryptanalyst's knowledge as follows:

Ciphertext-only attack: The enemy cryptanalyst has no information additional to intercepted ciphertexts;

Known-plaintext attack: The enemy cryptanalyst knows additionally some plaintext/ciphertext pairs for the current key.

Chosen-plaintext attack: The enemy cryptanalyst can obtain the ciphertexts for any specified plaintexts for the current key.

The chosen-plaintext attack is the most powerful of the above attacks. If a cipher is secure against the chosen-plaintext attack, then it is also secure against other attacks. In practice, one would like to use a cipher that is secure against a chosen-plaintext attack even if the enemy cryptanalyst would rarely have the chance to mount more than a ciphertext-only attack.

2.2.2 Unconditional and computational security

The security of a cipher system depends crucially on the computational capability of the enemy cryptanalyst. A cipher system is called *unconditionally secure* if it is secure against an enemy cryptanalyst with unlimited computational resources. Unconditional security, also called *theoretical security* [53], deals with the impossibility of breaking a cipher. A cipher that is secure against an enemy cryptanalyst with specified limited computational power is called *computationally secure.* Computational security, also called *practical security,* deals with the difficulty of breaking a cipher. All presently known unconditionally secure cipher systems are *impractical* for reasons to be discussed below. Moreover, no practical cipher has yet been *proved* to be computationally secure.

Unconditional security

Although in most applications unconditional security is not necessary and is impossible to achieve for a practical cipher, the study of unconditional security does give much insight and aids in the design and use of practical ciphers. For example, the basic motivation for a stream cipher [51] is the perfect secrecy provided by the "one-time pad" system.

Definition 2 *(Shannon 1949) A cipher provides perfect secrecy if the ciphertext and plaintext blocks are statistically independent.*

The achievability of perfect secrecy was shown by Shannon in his 1949 paper [53]. The following "one-time-key group cipher", described in Example 1, provides such perfect secrecy. The idea of using such a "one-time" key system was first proposed by Vernam [56] in 1926. The Vernam cipher is often called the "one-time pad" system. Although it was believed for a long time that the one-time pad is "unbreakable", the fact that it provides perfect secrecy was first *proved* by Shannon.

Example 1 (One-time-key group cipher) *Consider the block cipher shown in Fig.2.2 where $\otimes$ is a group operation defined on the set $\mathbb{F}_2^m$. This cipher has perfect secrecy if the key is chosen uniformly at random and independently for each plaintext block.*

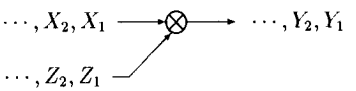


Figure 2.2: The one-time-key group cipher: the keys Z_i are chosen uniformly at random and independently.

A perfect secrecy system is usually impractical because, as Shannon showed, an unlimited amount of key will be needed if we allow messages of unlimited length. However, the idea of perfect secrecy establishes a well-known principle for cryptographic practice, viz., that for the utmost security *one should change the secret key as frequently as possible.*

Unconditional security can also be achieved by data compression. Shannon defined a cipher system to be *strongly ideal* if, for a fixed key, the sequence of ciphertext blocks give no information about the key. Shannon noted that if the plaintext has no redundancy, i.e., if all plaintext blocks are independent and uniformly random,

then almost every block cipher will be strongly ideal, that is, such a cipher system will be secure against a ciphertext-only attack even if the same key is used to encipher many plaintexts. Unfortunately, none of the presently known data processing techniques can achieve such perfect data-compression. But Shannon's work establishes another principle of cryptographic practice, viz., that for the utmost security *the plaintext data should be as random as possible.* This can be achieved either by data-compression or by homophonic substitution [27].

A strongly ideal system is, by definition, secure against a ciphertext-only attack. Against a known-plaintext (or chosen-plaintext) attack, not every strongly ideal system is secure. For example, consider the group block cipher of Fig.2.2. Even if the same key is used many times, this cipher system is strongly ideal when the plaintext blocks are independent and uniformly distributed. However, given one plaintext/ciphertext pair, one can easily determine the key. Thus, although this cipher is unconditionally secure against a ciphertext-only attack, it can be easily broken in a known-plaintext attack if the secret key is used more than once.

For a block cipher used in a "non-one-time" mode, i.e., when one key is used to encrypt many plaintext blocks, perfect-secrecy as defined by Shannon can never be achieved because identical ciphertext blocks imply identical plaintext blocks. The unconditional security against a known-plaintext (or chosen-plaintext) attack when the key is used more than once was considered by Massey in [35]. Massey considered *perfect secrecy against an order-L known-plaintext attack* for $0 \leq L \leq 2^m - 2$, i.e., in the case when the enemy cryptanalyst knows L distinct plaintext/ciphertext pairs for the current key but none of these ciphertexts agree with that under attack. In particular, if $L = 2^m - 2$, the cipher is "truly unbreakable". A block cipher is *truly unbreakable* if, for every integer i, $0 \leq i \leq 2^m - 2$, for every choice of distinct plaintexts $x_1, x_2, \ldots, x_i$ and distinct ciphertexts $y_1, y_2, \ldots, y_i$ such that the actual key is consistent with the plaintext/ciphertext pairs $(x_1, y_1), \ldots, (x_i, y_i)$, the plaintext X conditioned upon taking values in $\mathbb{F}_2^m - \{x_1, ..., x_i\}$ is independent of the ciphertext Y. In [35], it was shown that such a truly unbreakable block cipher can be achieved only by a "random" cipher.

Example 2 *A block cipher is truly unbreakable if and only if the key Z serves to choose the encryption function equally likely from the the set of all invertible functions from $\mathbb{F}_2^m$ to $\mathbb{F}_2^m$.*

The truly unbreakable cipher is impractical for two reasons. First, it follows from Example 2 and the fact that there are $(2^m)!$ invertible functions from $\mathbb{F}_2^m$ to $\mathbb{F}_2^m$ that the key space S_z must contain at least $(2^m)!$ different keys. Note that $\log_2(2^m!) \approx$

$(m - 1.44)2^m$ bits so that on average to transmit one bit of message one needs to transmit about 2^m key bits securely. Second, for large m, the implementation of a randomly chosen invertible function is infeasible in the practice. Shannon has shown [52] that to compute an invertible function from $\mathbb{F}_2^m$ to $\mathbb{F}_2^m$, at least 2^m binary operations are required for virtually all of the $(2^m)!$ such invertible functions.

In this section, we have considered three kinds of unconditional security for block ciphers within Shannon's model of a secrecy system.

Perfect secrecy provides the best possible protection against a ciphertext-only attack and can be achieved by a "one-time-pad" system. However, a perfect system is impractical because it requires one to transmit securely at least as much keys as plaintexts. The practical implication of perfect secrecy is that the user should change the secret key as often as possible.

A strongly ideal system is unconditionally secure against a ciphertext-only attack even when a small amount of key is used to encrypt a large amount of plaintext. A strongly ideal system requires essentially that the plaintext source be a true random number generator, which makes such a system impractical. The practical implication is that the plaintext data should be as random as possible.

An unbreakable cipher system provides unconditional security against a chosen-plaintext attack. It requires that the encryption function be a "random" invertible function. Although a truly unbreakable cipher is even more impractical than a perfect secrecy system, it indicates that a practical block cipher should be designed such that each encryption function appears to be a randomly chosen invertible function.

Computational security

In practice, no attacker can have unlimited computational power. The security of a practical cipher system depends therefore not on the theoretical impossibility of breaking a cipher, but rather on the practical difficulty of attacks. A cipher is computationally secure if the difficulty of an optimum attack exceeds the computational capability of the cryptanalyst. Shannon described such difficulty (for ciphertext-only attack) by the work characteristic $W(n)$ of the cipher, defined as the amount of work required for determining the key when n ciphertexts are known. One can also consider the work characteristic for other type of attacks. Throughout this work we will use the commonly used word "complexity" to describe such difficulty. The *complexity* of an attack is generally understood to mean the average number of operations used in the attack. Note that for a cipher to be computationally secure means that the complexity of the optimum attack exceeds the computational capability of the

enemy cryptanalyst. To prove that a cipher is computationally secure would imply a useful lower-bound on the complexity of solving a certain computational problem. This is at present impossible for virtually all computational problems. Therefore, in practice, the evaluation of the security of a cipher relies on the complexity of the best attack that is presently known. A practical block cipher is generally considered secure if none of the known attacks can do much better than the exhaustive key-search attack. In an *exhaustive key-search* ciphertext-only attack on a block cipher, each of the possible keys is tried in turn to decipher one (or more) intercepted ciphertext block(s) until one key results in "readable" plaintext block(s). The complexity of this attack can be described as the number of tried decryptions, which is, on average, 2^{k_t-1} for a block cipher with true key size k_t. Exhaustive key-search is a "brute-force" attack that can be applied to any block cipher. Thus, for a block cipher to be secure, its true key size must be large enough to make the exhaustive key-search attack infeasible.

2.2.3 Data- and processing-complexity of a specified attack

The complexity of an attack can be divided into two parts: data complexity and processing complexity. *Data-complexity* is the amount of input data needed for the attack while *processing-complexity* is the amount of computations needed to process such data. The dominant component is usually described as the complexity of the attack. For example, in an exhaustive key-search attack, the amount of input data needed for the attack is the number of intercepted ciphertext blocks (or the number of plaintext/ciphertext pairs, in the known-plaintext case), which is in general a very small number in comparison with the number of operations (2^{k_t-1} decryptions with different keys required on the average in the key-search) needed in the attack. Therefore, the complexity of such a key-search attack is actually the processing complexity. Another example is Biham and Shamir's differential cryptanalysis [6], which is a chosen-plaintext attack. For differential cryptanalysis, the complexity is dominated by the number of plaintext/ciphertext pairs needed in the attack, while the number of computations used in the attack is relatively small [9]. Therefore, the complexity of differential cryptanalysis is essentially the data complexity.

In general, for a block cipher with block length m bits and true key size k_t bits, the data complexity of a known-plaintext (or chosen-plaintext) attack can be measured by the number of known (chosen) plaintext/ciphertext pairs needed for the attack, which is at most 2^m, the total number of such pairs for a fixed key. The processing

complexity can be upper-bounded by 2^{k_t} encryptions because of the universality of the exhaustive key-search attack and because encryptions are usually efficiently computable. Thus, we can say that a cipher is *computationally secure* if none of the attacks on the cipher has data complexity significantly smaller than 2^m encryptions and processing complexity significantly smaller than 2^{k_t} encryptions. A cipher will be called *practically secure* against a specified attack if, for this attack, the data complexity is about 2^m plaintext/ciphertext pairs *or* the processing complexity is about 2^{k_t} encryptions. For the cryptanalyst, data-complexity is *passive* complexity, he must wait for the user to generate the plaintext/ciphertext pairs for him. On the other hand, processing complexity is *active* complexity and can be overcome by, say, using more powerful computers.

2.2.4 Block cipher parameters

Block length m

For a block cipher to be secure, its block length m must be large enough to deter statistical analysis [30], i.e., to deny the opponent any advantage that some blocks appear more often than the others. The choice of $m = 64$ bits as in the Data Encryption Standard (DES) [14] is generally conceded to be large enough to make frequency analysis infeasible for most applications. Moreover, the block length m should be such that, for a fixed key, the number of plaintext/ciphertext pairs that an attacker can obtain in practice is much smaller than 2^m, the number of such pairs that would completely describe the encryption function for a fixed key.

When the block length of a cipher becomes large, the complexity of implementation also grows. Because the complexity of implementing a randomly chosen invertible function increases exponentially with the block length [52, 58], only those "simple" functions that *appear* to be random have a chance to serve as practical encryption functions when the block length m is large. However, as Shannon has pointed out in [53], the ease of computation of the encryption function $E(\cdot, z)$ and the decryption $D(\cdot, z)$ for all z does not imply that the solving for key z from the equations $y = E(x, z)$ and $x = D(y, z)$ will be easy when x and y are known.

Key length k and true key size k_t

For a block cipher to be secure against exhaustive key-search attack, the true key size k_t of the key should be large enough so that the 2^{k_t-1} encryptions needed for key-search attack is far beyond the enemy cryptanalyst's capability. On the other hand, the key-length k should be as small as possible so that the generation, distribution

and storage of the key can be carried out efficiently and securely. For example, DES has a key length 64 bits, but the true key size is only 56 bits. An exhaustive key-search attack on DES is beyond feasibility, but not too far beyond. Almost from the beginning, DES has been criticized for its short key. Diffie and Hellman [20] have proposed a conceptual special-purpose computer that would cost 20 million dollars and that would break DES by exhaustive key-search in 12 hours. Many suggestions have been made to increase artificially the true key size for DES [5, 20]. For example, extending the DES true key size to about 128 bits by triple encryption is a standard way of using DES [2, 55].

2.3 Design Principles for Block Ciphers

A good block cipher should be "hard to break and easy to implement". Both the encryption function $E(\cdot, z)$ and the decryption function $D(\cdot, z)$ must be easily computable. To solve for the key z from $y = E(x, z)$ and $x = D(y, z)$, however, should be a difficult problem. The design principles for a block cipher can be correspondingly divided into implementation principles and security principles.

2.3.1 General design principles for security

The only two generally accepted design principles for practical ciphers are the principles of confusion and diffusion that were suggested by Shannon in [53].

Confusion: To design a cipher according to the principle of confusion means that one designs it so as "to make the the relation between the simple statistics of ciphertext and the simple description of key a very complex and involved one" [53]. We state the principle of confusion as *the dependence of the key on the plaintext and ciphertext should be so complex that it is useless for cryptanalysis.* For example, the binary equations that describe the block cipher should be so "nonlinear" and "complex" that to solve for z from x and $y = E(x, z)$ is infeasible.

Diffusion: To design a cipher according to the principle of diffusion means that one designs it to ensure that "the statistical structure of plaintext which leads to its redundancy is 'dissipated' into long term statistics" [53]. We state the principle of diffusion as follows: *For virtually every key, the encryption function should be such that there is no statistical dependence between simple structures in the plaintext and simple structures in the ciphertext and that there is no simple relation between*

different encryption functions. The principle of diffusion requires, for instance, that a block cipher should be designed to be *"complete"* [28], i.e., each bit of plaintext and each bit of key should influence each bit of ciphertext.

2.3.2 Design principles for implementation

A block cipher can be implemented in software or hardware. Hardware implementation, which is usually done by a dedicated VLSI-chip, can achieve high speed. Software implementation has the advantage of flexibility and low cost. Based on the different properties of software and hardware, the design principles for a block cipher can be divided according to the intended method of implementation.

Design principles for software implementation

The following principles are proposed:

Using subblocks. Cipher operations should operate on subblocks whose length is "natural" for software, e.g., 8, 16 or 32 bits. Bitwise permutations that are hard to do in software should be avoided.

Using simple operations. The cipher operations on subblocks should be easily implemented with basic instructions for standard processors such as addition, multiplication, shifting and so on.

Design principles for hardware implementation

Similarity of encryption and decryption. The encryption and decryption processes should differ essentially only in the way of using the secret key so that the same device can be used for both encryption and decryption.

Regular structure. The cipher should have a regular modular structure to facilitate VLSI implementation.

2.4 Iterated Ciphers

2.4.1 Iterated cipher and round function

A block cipher is called an *iterated cipher* if it is based on iterating a simple function f several times as shown in Fig.2.3. Each iteration is called a *round.* The output of

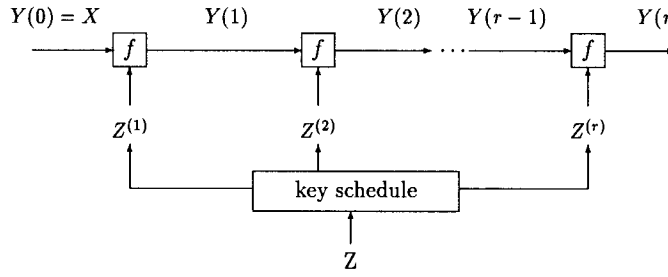


Figure 2.3: An r-round iterated cipher with round function f.

each round is a function of the output of the previous round and of a subkey derived from the full secret key by a key-schedule algorithm. Such a secret-key block cipher with r-iterations is called an r-round iterated cipher. The function f is called the *round function.* For example, DES is a 16-round iterated cipher.

The method of iteration is used in block cipher design [4, 42] because it is consistent with all the basic design principles mentioned before. A simple round function can be efficiently implemented, while the iteration of a suitably chosen round function can provide the necessary confusion and diffusion. A good example can be found in chapter 4 where we shall prove that, in the differential cryptanalysis of Markov ciphers, the data-complexity of the attack will increase *exponentially* with the number of iterations whereas the complexity of implementation increases only *linearly.*

2.4.2 Constructions of E/D similar iterated ciphers

In this section, we shall discuss some constructions of iterated ciphers that have similarity of encryption and decryption (E/D similar). These ciphers are based on the following transformations:

Involution cipher. A function $In(\cdot,\cdot)$ from $\mathbb{F}_2^m \times \mathbb{F}_2^k$ to $\mathbb{F}_2^m$ will be called an *involution* cipher if for every choice of key z, $In(\cdot, z)$ is an involution on $\mathbb{F}_2^m$, i.e., if for every z in $\mathbb{F}_2^k$, $In(In(x,z),z) = x$ for all x in $\mathbb{F}_2^m$.

Group cipher. A cipher will be called a *group cipher* if the ciphertext Y is computed from plaintext X and key Z as $Y = X \otimes Z$ where $\otimes$ is a group operation for $\mathbb{F}_2^m$. Note that the key must take values in $\mathbb{F}_2^m$ for a group cipher, i.e., $k = m$. Note also that X can be computed from Y as $X = Y \otimes Z^{-1}$, where Z^{-1} denotes the group inverse of Z.

Involutary permutation. An involutary permutation is an involution $P_I(\cdot)$ on the set $\mathbb{F}_2^m$, i.e., $P_I(P_I(x)) = x$ for all x in $\mathbb{F}_2^m$. For example, "swapping" of halves of x is an involutary permutation.

I. Using an involution cipher only

If the round function is an involution cipher $In(X, Z)$, then the iterated cipher shown in Fig.2.4 is encryption/decryption (E/D) similar:

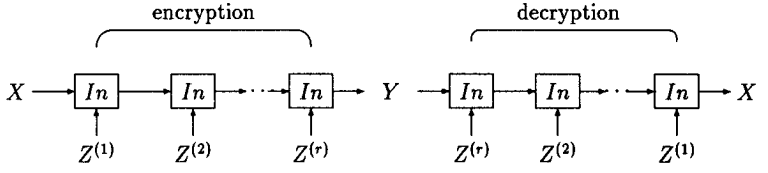


Figure 2.4: Illustration of the similarity of encryption and decryption for an iterated cipher of Type I with an involution cipher as round function.

The involution-cipher-only construction has the obvious disadvantage that, for an even number of rounds, the choice $Z^{(2i-1)} = Z^{(2i)}$ for all i causes the resulting encryption function to be the identity. To overcome such a "cancellation effect", one can insert involutary permutations.

II. Using an involution cipher and an involutary permutation

If the round function is an involution cipher, $In(X, Z)$, followed by a key independent involutary permutation P_I, i.e., if $f(X, Z) = P_I(In(X, Z))$, then the iterated cipher shown in Fig.2.5 is E/D similar. Note that the additional permutation inserted after the last round just undoes the permutation of the last round. The decryption is done by using the subkeys $Z^{(i)}$ in reverse order. The block cipher DES [14] and the DES-like ciphers FEAL [54] and LOKI [11] are of this type.

III. Using a group cipher and an involution

The round function f is a group cipher $X \otimes Z_A$ followed by an involution cipher $In(X, Z_B)$, i.e.,

$$f(X, Z) = In(X \otimes Z_A, Z_B). \tag{2.1}$$

The iterated cipher based on this round function is shown in Fig.2.6. An extra group cipher is put at the end of the last round of the encryption process. It is easy to

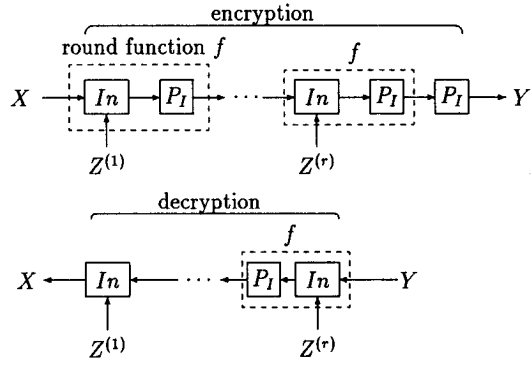


Figure 2.5: Illustration of the similarity of encryption and decryption for an iterated cipher of Type II whose round function is an involution cipher followed by involutary permutation.

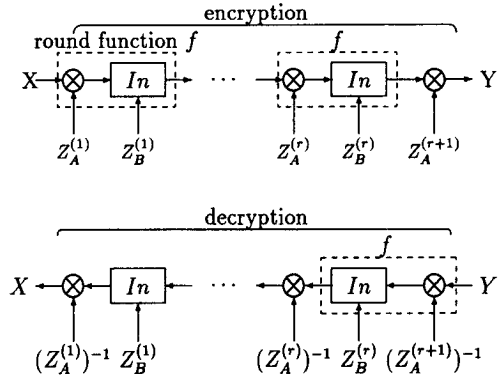


Figure 2.6: Illustration of the similarity of encryption and decryption for an iterated cipher of Type III whose round function is a group cipher followed by an involution cipher.

see from Fig.2.6 that this cipher is E/D similar when the encryption subkeys are $Z_A^{(1)}, Z_B^{(1)}, Z_A^{(2)}, Z_B^{(2)}, \ldots, Z_A^{(r)}, Z_B^{(r)}, Z_A^{(r+1)}$ and the decryption subkeys are $(Z_A^{(r+1)})^{-1}$, $Z_B^{(r)}$, $(Z_A^{(r)})^{-1}$, $Z_B^{(r-1)}, \ldots, (Z_A^{(2)})^{-1}$, $Z_B^{(1)}, (Z_A^{(1)})^{-1}$. Our block cipher PES [31] is of this type.

IV. Using a group cipher, an involution cipher and an involutary permutation

For this construction, the round function f has the form

$$f(X, Z) = P_I(In(X \otimes Z_A, Z_B)) \tag{2.2}$$

where $X \otimes Z_A$ is a group cipher, which is followed by an involution cipher $\mathrm{In}(X, Z_B)$, and where the involutary permutation P_I is an automorphism of the group $(\mathbb{F}_2^m, \otimes)$, i,e., $P_I(a \otimes b) = P_I(a) \otimes P_I(b)$ for all a and b in $\mathbb{F}_2^m$. The iterated cipher based on this round function is shown in Fig.2.7. Note that an additional involutary permutation and group cipher are put at the end of the encryption process. Note also that the two consecutive involutary permutations shown at the end of the last round cancel one another. The IDEA cipher we will consider in the following chapters is of this type.

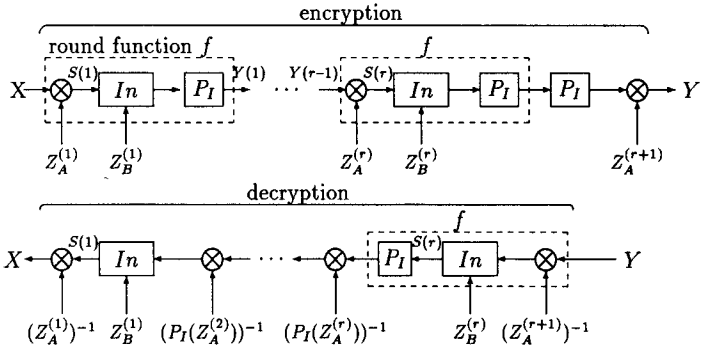


Figure 2.7: Illustration of the similarity of encryption and decryption for an iterated cipher of Type IV whose round function is a group cipher followed by an involution cipher and an involutary permutation.

Theorem 1 *The iterated cipher, whose round function has form (2.2) and which is followed by an additional involutary permutation and a group cipher, has similarity of encryption and decryption if, for the encryption subkeys* $Z_A^{(1)}, Z_B^{(1)}, Z_A^{(2)}, \ldots,$

$Z_A^{(r)}, Z_B^{(r)}, Z_A^{(r+1)}$, *one uses the decryption subkeys* $K_A^{(1)}, K_B^{(1)}, K_A^{(2)}, \ldots, K_A^{(r)}, K_B^{(r)}, K_A^{(r+1)}$ *computed from the encryption subkeys as*

$$\begin{array}{rcll} K_A^{(i)} & = & (P_I(Z_A^{(r-i+2)}))^{-1} & \text{for } i = 2, 3, ..., r, \\ K_A^{(i)} & = & (Z_A^{(r-i+2)})^{-1} & \text{for } i = 1, r+1, \\ K_B^{(i)} & = & Z_B^{(r-i+1)} & \text{for } i = 1, 2, ..., r. \end{array} \tag{2.3}$$

Proof. In the encryption process for $1 \leq i < r$, we have

$$\begin{aligned} S(i+1) &= P_I\left(In(S(i), Z_B^{(i)})\right) \otimes Z_A^{(i+1)} \\ &= P_I\left(In(S(i), Z_B^{(i)}) \otimes P_I(Z_A^{(i+1)})\right), \end{aligned}$$

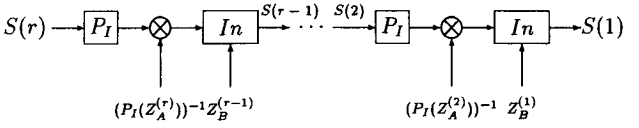
where we have used the fact that the involution P_I is also an automorphism for $(\mathbb{F}_2^m, \otimes)$. Thus,

$$P_I(S(i+1)) \otimes (P_I(Z_A^{(i+1)}))^{-1} = In(S(i), Z_B^{(i)})$$

and thus also

$$In\left(P_I(S(i+1)) \otimes (P_I(Z_A^{(i+1)}))^{-1}, Z_B^{(i)}\right) = S(i).$$

This shows that $S(1)$ can be obtained from $S(r)$ by using the process:



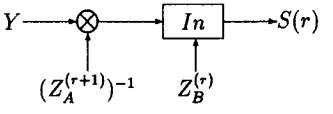
But in the encryption process,

$$Y = In(S(r), Z_B^{(r)}) \otimes Z_A^{(r+1)}$$

so that

$$S(r) = In(Y \otimes (Z_A^{(r+1)})^{-1}, Z_B^{(r)}).$$

Thus $S(r)$ can be obtained from Y by the process:



Finally, in the encryption process,

$$S(1) = X \otimes Z_A^{(1)}$$

so that X can be obtained from $S(1)$ by the process: $X = S(1) \otimes (Z_A^{(1)})^{-1}$.

Combining these three processes gives the complete decryption process shown in Fig.2.7, which we see is precisely the same process as the encryption process except for the choice of subkeys, as was to be shown. □

Chapter 3

The Block Cipher IDEA

The block cipher IDEA (for International Data Encryption Algorithm) was first presented by us in [32]; its previous version PES (for Proposed Encryption Standard) was proposed in [31]. In both ciphers, the plaintext and the ciphertext are 64 bit blocks, while the secret key is 128 bits long. Both ciphers were based on the new design concept of "mixing operations from different algebraic groups". The required "confusion" was achieved by successively using three "incompatible" group operations on pairs of 16-bit subblocks and the cipher structure was chosen to provide the necessary "diffusion". The cipher structure was further chosen to facilitate both hardware and software implementations. The IDEA cipher is an improved version of PES and was developed to increase the security against differential cryptanalysis.

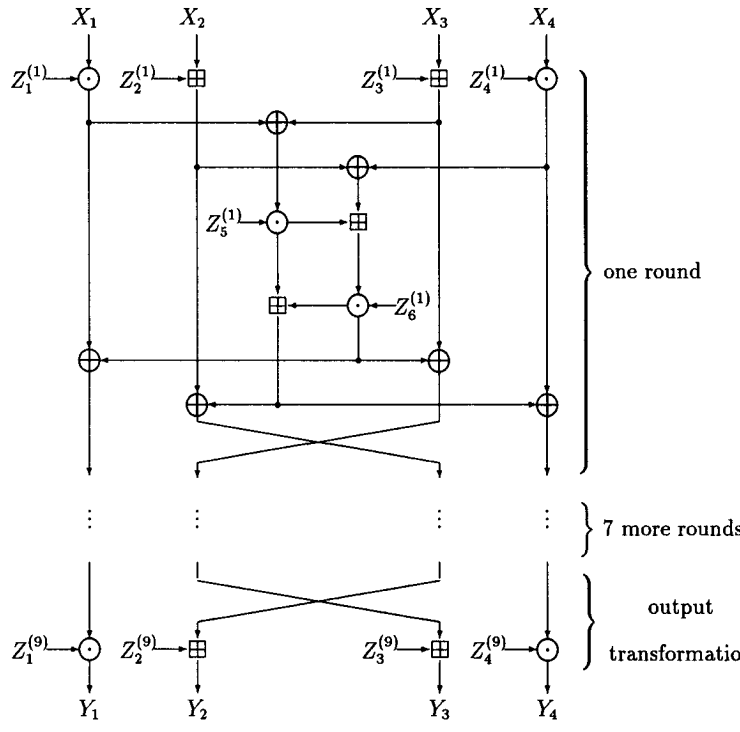
3.1 Description of IDEA

The cipher IDEA is an iterated cipher consisting of 8 rounds followed by an output transformation. The complete first round and the output transformation are depicted in the computational graph shown in Fig.3.1.

3.1.1 The encryption process

In the encryption process shown in Fig.3.1, three different group operations on pairs of 16-bit subblocks are used, namely,

- bit-by-bit exclusive-OR of two 16-bit subblocks, denoted as $\oplus$;
- addition of integers modulo 2^{16} where the 16-bit subblock is treated as the usual radix-two representation of an integer; the resulting operation is denoted as $\boxplus$;



X_i : 16-bit plaintext subblock
Y_i : 16-bit ciphertext subblock
$Z_i^{(r)}$: 16-bit key subblock
$\oplus$: bit-by-bit exclusive-OR of 16-bit subblocks
$\boxplus$: addition modulo 2^{16} of 16-bit integers
$\odot$: multiplication modulo $2^{16}+1$ of 16-bit integers with the zero subblock corresponding to 2^{16}

Figure 3.1: Computational graph for the encryption process of the IDEA cipher.

- multiplication of integers modulo $2^{16}+1$ where the 16-bit subblock is treated as the usual radix-two representation of an integer except that the all-zero subblock is treated as representing 2^{16}; the resulting operation is denoted as $\odot$.

As an example of these group operations, note that

$$(0,...,0)\odot(1,0,...,0) = (1,0,...,0,1)$$

because

$$2^{16}2^{15} \bmod (2^{16}+1) = 2^{15}+1.$$

The 64-bit plaintext block X is partitioned into four 16-bit subblocks X_1, X_2, X_3, X_4, i.e., $X = (X_1, X_2, X_3, X_4)$. The four plaintext subblocks are then transformed into four 16-bit ciphertext subblocks Y_1, Y_2, Y_3, Y_4 [i.e., the ciphertext block is $Y = (Y_1, Y_2, Y_3, Y_4)$] under the control of 52 key subblocks of 16 bits that are formed from the 128-bit secret key in a manner to be described below. For $r = 1, 2, \ldots, 8$, the six key subblocks used in the r-th round will be denoted as $Z_1^{(r)}, .., Z_6^{(r)}$. Four 16-bit key subblocks are used in the output transformation; these subblocks will be denoted as $Z_1^{(9)}$, $Z_2^{(9)}$, $Z_3^{(9)}$, $Z_4^{(9)}$.

3.1.2 The decryption process

The computational graph of the decryption process is essentially the same as that of the encryption process (cf. Sec.3.4.1), the only change being that the decryption key subblocks $K_i^{(r)}$ are computed from the encryption key subblocks $Z_i^{(r)}$ as follows:

$$\begin{aligned}
(K_1^{(r)}, K_2^{(r)}, K_3^{(r)}, K_4^{(r)}) &= (Z_1^{(10-r)^{-1}}, -Z_3^{(10-r)}, -Z_2^{(10-r)}, Z_4^{(10-r)^{-1}}) \text{ for } r=2,3,..,8;\\
(K_1^{(r)}, K_2^{(r)}, K_3^{(r)}, K_4^{(r)}) &= (Z_1^{(10-r)^{-1}}, -Z_2^{(10-r)}, -Z_3^{(10-r)}, Z_4^{(10-r)^{-1}}) \text{ for } r=1 \text{ and } 9;\\
(K_5^{(r)}, K_6^{(r)}) &= (Z_5^{(r)}, Z_6^{(r)}) \qquad\qquad \text{for } r=1,2,..,8;
\end{aligned}$$

where Z^{-1} denotes the multiplicative inverse (modulo $2^{16}+1$) of Z, i.e., $Z \odot Z^{-1} = 1$ and $-Z$ denotes the additive inverse (modulo 2^{16}) of Z, i.e., $-Z \boxplus Z = 0$.

The computation of decryption key subblocks from the encryption key subblocks is also shown in table 3.1.

3.1.3 The key schedule

The 52 key subblocks of 16 bits used in the encryption process are generated from the 128-bit user-selected key as follows: The 128-bit user-selected key is partitioned into 8 subblocks that are directly used as the first eight key subblocks, where the ordering of the key subblocks is defined as follows: $Z_1^{(1)}, Z_2^{(1)}, .., Z_6^{(1)}, Z_1^{(2)}, .., Z_6^{(2)}, ..,$

Encryption key subblocks		Decryption key subblocks	
1-st round	$Z_1^{(1)} Z_2^{(1)} Z_3^{(1)} Z_4^{(1)}$ $Z_5^{(1)} Z_6^{(1)}$	1-st round	$Z_1^{(9)^{-1}} - Z_2^{(9)} - Z_3^{(9)} Z_4^{(9)^{-1}}$ $Z_5^{(8)} Z_6^{(8)}$
2-nd round	$Z_1^{(2)} Z_2^{(2)} Z_3^{(2)} Z_4^{(2)}$ $Z_5^{(2)} Z_6^{(2)}$	2-nd round	$Z_1^{(8)^{-1}} - Z_3^{(8)} - Z_2^{(8)} Z_4^{(8)^{-1}}$ $Z_5^{(7)} Z_6^{(7)}$
3-rd round	$Z_1^{(3)} Z_2^{(3)} Z_3^{(3)} Z_4^{(3)}$ $Z_5^{(3)} Z_6^{(3)}$	3-rd round	$Z_1^{(7)^{-1}} - Z_3^{(7)} - Z_2^{(7)} Z_4^{(7)^{-1}}$ $Z_5^{(6)} Z_6^{(6)}$
4-th round	$Z_1^{(4)} Z_2^{(4)} Z_3^{(4)} Z_4^{(4)}$ $Z_5^{(4)} Z_6^{(4)}$	4-th round	$Z_1^{(6)^{-1}} - Z_3^{(6)} - Z_2^{(6)} Z_4^{(6)^{-1}}$ $Z_5^{(5)} Z_6^{(5)}$
5-th round	$Z_1^{(5)} Z_2^{(5)} Z_3^{(5)} Z_4^{(5)}$ $Z_5^{(5)} Z_6^{(5)}$	5-th round	$Z_1^{(5)^{-1}} - Z_3^{(5)} - Z_2^{(5)} Z_4^{(5)^{-1}}$ $Z_5^{(4)} Z_6^{(4)}$
6-th round	$Z_1^{(6)} Z_2^{(6)} Z_3^{(6)} Z_4^{(6)}$ $Z_5^{(6)} Z_6^{(6)}$	6-th round	$Z_1^{(4)^{-1}} - Z_3^{(4)} - Z_2^{(4)} Z_4^{(4)^{-1}}$ $Z_5^{(3)} Z_6^{(3)}$
7-th round	$Z_1^{(7)} Z_2^{(7)} Z_3^{(7)} Z_4^{(7)}$ $Z_5^{(7)} Z_6^{(7)}$	7-th round	$Z_1^{(3)^{-1}} - Z_3^{(3)} - Z_2^{(3)} Z_4^{(3)^{-1}}$ $Z_5^{(2)} Z_6^{(2)}$
8-th round	$Z_1^{(8)} Z_2^{(8)} Z_3^{(8)} Z_4^{(8)}$ $Z_5^{(8)} Z_6^{(8)}$	8-th round	$Z_1^{(2)^{-1}} - Z_3^{(2)} - Z_2^{(2)} Z_4^{(2)^{-1}}$ $Z_5^{(1)} Z_6^{(1)}$
output transform.	$Z_1^{(9)} Z_2^{(9)} Z_3^{(9)} Z_4^{(9)}$	output transform.	$Z_1^{(1)^{-1}} - Z_2^{(1)} - Z_3^{(1)} Z_4^{(1)^{-1}}$

Table 3.1: The encryption and decryption key subblocks.

$Z_1^{(8)},..,Z_6^{(8)}$, $Z_1^{(9)},Z_2^{(9)},Z_3^{(9)},Z_4^{(9)}$. The 128-bit user-selected key is then cyclic shifted to the left by 25 positions, after which the resulting 128-bit block is again partitioned into eight subblocks that are taken as the next eight key subblocks. The obtained 128-bit block is again cyclic shifted to the left by 25 positions to produce the next eight key subblocks, and this procedure is repeated until all 52 key subblocks have been generated.

3.2 Group Operations and their Interaction

The IDEA cipher is based on the new design concept of mixing operations from different algebraic groups having the same number of elements. Group operations were chosen because the statistical relation of any three random variables U, V, W related by a group operation as $W = U * V$ has the "perfect secrecy" property that if any one of the three random variables is chosen independently of the others and equally likely to be any group element, then the other two random variables are statistically independent. The interaction of the different group operations contributes to the "confusion" required for a secure cipher, as will be explained in the following two sections.

The interaction of the different group operations will now be considered in terms of isotopism of quasigroups and in terms of polynomial expressions. To generalize the discussion beyond the case of 16-bit subblocks, let n be one of the integers 1,2,4,8 or 16 so that the integer $2^n + 1$ is a prime, and let $\mathbf{Z}_{2^n}$ denote the ring of integers modulo 2^n. Let $(\mathbf{Z}_{2^n+1}^*, \cdot)$ denote the multiplicative group of the non-zero elements of the field $\mathbf{Z}_{2^n+1}$, let $(\mathbf{Z}_{2^n}, +)$ denote the additive group of the ring $\mathbf{Z}_{2^n}$, and let $(\mathbf{F}_2^n, \bigoplus)$ denote the group of n-tuples over $\mathbf{F}_2$ under the bitwise exclusive-or operation. Define the *direct* mapping d from $\mathbf{Z}_{2^n+1}^*$ onto $\mathbf{Z}_{2^n}$ as

$$d(i) = i \text{ for } i \neq 2^n \text{ and } d(2^n) = 0. \tag{3.1}$$

3.2.1 The three operations as quasigroup operations

Let S be a non-empty set and let $*$ denote an operation from pairs (a, b) of elements of S to an element $a * b$ of S. Then $(S, *)$ is said to be a *quasigroup* if, for all a and b in S, the equations $a * x = b$ and $y * a = b$ both have exactly one solution in S. A *group* is a quasigroup in which the operation is associative, i.e., for which $a * (b * c) = (a * b) * c$ for all a, b and c in S. The quasigroups $(S_1, *_1)$ and $(S_2, *_2)$ are said to be *isotopic* if there are bijective mappings $\theta, \phi, \psi : S_1 \to S_2$, such that,

$$\theta(x) *_2 \phi(y) = \psi(x *_1 y) \qquad \text{for all } x \text{ and } y \text{ in } S_1.$$

Such a triple (θ, ϕ, ψ) of bijections is called an *isotopism* of $(S_1, *_1)$ onto $(S_2, *_2)$. Two groups are said to be *isomorphic* if they are isotopic as quasigroups and the isotopism has the form (θ, θ, θ). It can be shown that two groups are isomorphic if and only if they are isotopic [18]. Note that every isomorphism between two groups is also an isotopism, but the converse is not true in general. In general for two isomorphic groups, there will be many more isotopisms between these groups than there will be isomorphisms. For this reason, we consider isotopisms rather than isomorphisms although our objects are all groups. The following theorem states some "incompatibility" properties of the three groups $(\mathbb{F}_2^n, \bigoplus)$, $(\mathbb{Z}_{2^n}, +)$ and $(\mathbb{Z}_{2^n+1}^*, \cdot)$ when $n \geq 2$.

Theorem 2 *For $n \in \{1, 2, 4, 8, 16\}$:*
1) The quasigroups $(\mathbb{F}_2^n, \bigoplus)$ and $(\mathbb{Z}_{2^n}, +)$ are not isotopic for $n \geq 2$.
2) The quasigroups $(\mathbb{F}_2^n, \bigoplus)$ and $(\mathbb{Z}_{2^n+1}^*, \cdot)$ are not isotopic for $n \geq 2$.
3) The triple (θ, ϕ, ψ) of bijections from $\mathbb{Z}_{2^n+1}^*$ to $\mathbb{Z}_{2^n}$ is an isotopism of $(\mathbb{Z}_{2^n+1}^*, \cdot)$ onto $(\mathbb{Z}_{2^n}, +)$ if and only if there exist c_1 and c_2 in $\mathbb{Z}_{2^n}$ and a generator α of the cyclic group $\mathbb{Z}_{2^n+1}^*$ such that, for all x in $\mathbb{Z}_{2^n+1}^*$,

$$\theta(x) - c_1 = \phi(x) - c_2 = \psi(x) - (c_1 + c_2) = \log_\alpha(x), \tag{3.2}$$

i.e., any isotopism between these groups is essentially the discrete logarithm. Moreover, when $n \geq 2$, none of the three bijections in an isotopism (θ, ϕ, ψ) from $\mathbb{Z}_{2^n+1}^*$ onto $\mathbb{Z}_{2^n}$ can be the direct mapping d defined in (3.1).

Proof.
1) For $n \geq 2$, the groups $(\mathbb{F}_2^n, \bigoplus)$ and $(\mathbb{Z}_{2^n}, +)$ are not isomorphic because $(\mathbb{Z}_{2^n}, +)$ is a cyclic group while $(\mathbb{F}_2^n, \bigoplus)$ is not. Thus, they are not isotopic as quasigroups.
2) $(\mathbb{Z}_{2^n+1}^*, \cdot)$ and $(\mathbb{Z}_{2^n}, +)$ are isomorphic groups for $n = 1, 2, 4, 8, 16$ because both groups are cyclic. Thus, $(\mathbb{Z}_{2^n+1}^*, \cdot)$ is isotopic to $(\mathbb{F}_2^n, \bigoplus)$ if and only if $(\mathbb{Z}_{2^n}, +)$ is isotopic to $(\mathbb{F}_2^n, \bigoplus)$, which is not the case for $n \geq 2$.
3) Suppose that (θ, ϕ, ψ) satisfies (3.2) for all x in $\mathbb{Z}_{2^n+1}^*$, then for every x and y in $\mathbb{Z}_{2^n+1}^*$,

$$\psi(x \cdot y) = \log_\alpha(x \cdot y) + c_1 + c_2 = \log_\alpha(x) + \log_\alpha(y) + c_1 + c_2 = \theta(x) + \phi(y).$$

Thus, (θ, ϕ, ψ) is indeed an isotopism.

Conversely, if (θ, ϕ, ψ) is an isotopism from $(\mathbb{Z}_{2^n+1}^*, \cdot)$ onto $(\mathbb{Z}_{2^n}, +)$, then for all x and y in $\mathbb{Z}_{2^n+1}^*$, $\theta(x) + \phi(y) = \psi(x \cdot y)$. Let $\theta_1(x) = \theta(x) - \theta(1)$, $\phi_1(x) = \phi(x) - \phi(1)$ and $\psi_1(x) = \psi(x) - \psi(1)$, then $(\theta_1, \phi_1, \psi_1)$ is also an isotopism from $(\mathbb{Z}_{2^n+1}^*, \cdot)$ onto

$(\mathbb{Z}_{2^n}, +)$ as is easily checked. Moreover, $\theta_1(1) = \phi_1(1) = \psi_1(1) = 0$. In the isotopism equation

$$\theta_1(x) + \phi_1(y) = \psi_1(x \cdot y), \tag{3.3}$$

setting x to 1 results in $\phi_1(y) = \psi_1(y)$ for all y in $\mathbb{Z}^*_{2^n+1}$, and then setting y to 1 in (3.3) results in $\theta_1(x) = \psi_1(x)$ for all x in $\mathbb{Z}^*_{2^n+1}$. Thus, the three mappings θ_1, ϕ_1 and ψ_1 are identical. Equation (3.3) can thus be written as

$$\psi_1(x \cdot y) = \psi_1(x) + \psi_1(y). \tag{3.4}$$

Let α be the element of $\mathbb{Z}^*_{2^n+1}$ such that $\psi_1(\alpha) = 1$, then (3.4) implies that $\psi_1(\alpha^i) = i$ for $i = 1, 2, ..2^n - 1$ and $\psi_1(\alpha^{2^n}) = 0$. This implies that α is a generator of the cyclic group $(\mathbb{Z}^*_{2^n+1}, \cdot)$ and that $\psi_1(x) = \log_\alpha(x)$ for all x in $\mathbb{Z}^*_{2^n+1}$. Letting $c_1 = \theta(1)$ and $c_2 = \phi(1)$, we arrive at (3.2).

Finally, if (θ, ϕ, ψ) is an isotopism from $(\mathbb{Z}^*_{2^n+1}, \cdot)$ onto $(\mathbb{Z}_{2^n}, +)$ and one of the mappings θ, ϕ and ψ is the direct mapping d, then there exist c in $\mathbb{Z}_{2^n}$ and α in $\mathbb{Z}^*_{2^n+1}$ such that

$$d(x) = \log_\alpha(x) + c \text{ for all } x \text{ in } \mathbb{Z}^*_{2^n+1}. \tag{3.5}$$

But then $d(1) = 1$ implies that $1 = \log_\alpha(1) + c = c$ so that $d(x) = \log_\alpha(x) + 1$. Moreover, for $n \geq 2$, $d(2) = 2$, which implies that $2 = \log_\alpha(2) + 1$ so that $\alpha = 2$. But then $d(2^n) = 0$ implies that $\log_2(2^n) + 1 = n + 1 = 0$ which is a contradiction because $n < 2^n - 1$ for $n \geq 2$. Thus, none of the mappings θ, ϕ and ψ can be the direct mapping d if $n \geq 2$. □

3.2.2 Polynomial expressions for multiplication and addition

In the encryption process of the cipher IDEA, multiplication modulo $2^n + 1$ and addition modulo 2^n are related via the direct mapping d and its inverse d^{-1}. More precisely, multiplication modulo $2^n + 1$ induces the function $g : \mathbb{Z}_{2^n} \times \mathbb{Z}_{2^n} \to \mathbb{Z}_{2^n}$ defined by

$$g(x, y) = d[(d^{-1}(x) \cdot d^{-1}(y)) \bmod (2^n + 1)] \quad \text{for all } x \text{ and } y \text{ in } \mathbb{Z}_{2^n}. \tag{3.6}$$

Note that $g(x, y)$ is the function that we denoted as $x \odot y$ in Section 3.1.1. Similarly, addition modulo 2^n (the operation $\boxplus$) induces a function $f^* : \mathbb{Z}^*_{2^n+1} \times \mathbb{Z}^*_{2^n+1} \to \mathbb{Z}^*_{2^n+1}$ defined as

$$f^*(x, y) = d^{-1}[(d(x) + d(y)) \bmod 2^n] \quad \text{for all } x \text{ and } y \text{ in } \mathbb{Z}^*_{2^n+1}. \tag{3.7}$$

We can and do extend the function f^* to a function $f : \mathbf{Z}_{2^n+1} \times \mathbf{Z}_{2^n+1} \rightarrow \mathbf{Z}_{2^n+1}$ as follows:

$$f(x,y) = \begin{cases} d^{-1}[(d(x)+d(y)) \bmod 2^n] & \text{for all } x \text{ and } y \text{ in } \mathbf{Z}^*_{2^n+1} \\ 0 & \text{otherwise.} \end{cases} \tag{3.8}$$

For example, when $n = 1$, the function f induced by addition modulo 2 is

$$f(x,y) = 2xy \bmod 3 \quad \text{for all } x \text{ and } y \text{ in } \mathbf{Z}_3.$$

Similarly, the function g induced by multiplication modulo 3 is

$$g(x,y) = x + y + 1 \bmod 2 \quad \text{for all } x \text{ and } y \text{ in } \mathbf{Z}_2.$$

In what follows in this section, we show the "nonlinearity" of the function f over the field $\mathbf{Z}_{2^n+1}$ and the "nonlinearity" of the function g over the ring $\mathbf{Z}_{2^n}$ in terms of their polynomial expressions when $n \geq 2$.

Theorem 3 *For $n \in \{2,4,8,16\}$:*
For every a in $\mathbf{Z}_{2^n+1} - \{0, 2^n\}$, the function $f(a,y)$ is a polynomial in y over the field $\mathbf{Z}_{2^n+1}$ with degree $2^n - 1$. Similarly, for every a in $\mathbf{Z}_{2^n+1} - \{0, 2^n\}$, the function $f(x,a)$ is a polynomial in x over $\mathbf{Z}_{2^n+1}$ with degree $2^n - 1$.

Example 3 *For $n = 2$, the function $f(x,y)$ over $\mathbf{Z}_5$ induced by addition modulo 4 is*

$$f(x,y) = 3(x^3y^2 + x^2y^3) + 3(x^3y + xy^3) + 2x^2y^2 + 4(x^2y + xy^2).$$

Proof of Theorem 3. For any finite field $\mathbb{F} = GF(q)$ and for every α in $\mathbb{F}^* = GF(q) - \{0\}$,

$$(-\alpha) \prod_{\beta \in \mathbb{F}^* - \{\alpha\}} (x - \beta) = \begin{cases} 1 & x = \alpha \text{ or } x = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.9}$$

as follows from the fact that, in any finite field, the product of all non-zero elements equals -1 so that

$$(-\alpha) \prod_{\beta \in \mathbb{F}^* - \{\alpha\}} (\alpha - \beta) = - \prod_{\beta \in \mathbb{F}^*} \beta = 1.$$

Thus, every function $h(\cdot)$ from $\mathbb{F}$ to $\mathbb{F}$ can be written as a polynomial over $\mathbb{F}$ of degree at most $q - 1$ as follows:

$$h(x) = \sum_{\alpha \in \mathbb{F}^*} h(\alpha)(-\alpha) \prod_{\beta \in \mathbb{F}^* - \{\alpha\}} (x - \beta) + (x^{q-1} - 1)[h(0) - \sum_{\alpha \in \mathbb{F}^*} h(\alpha)]. \tag{3.10}$$

Note that $f(0,y) = 0$ for all y in $\mathbb{F}$, and that $f(a,\cdot)$ for every $a \neq 0$ is a bijection from $\mathbb{F}^*$ to $\mathbb{F}^*$, so that

$$\sum_{\alpha \in \mathbb{F}^*} f(a,\alpha) = \sum_{\alpha \in \mathbb{F}^*} \alpha = 0 \quad \text{for every } a \neq 0 \text{ and for } \mathbb{F} \neq GF(2).$$

From the definition of $f(x,y)$ and from equation (3.10), the function $f(a,y)$ can be written for every $a \neq 0$ as

$$\begin{aligned} f(a,y) &= \begin{cases} a+y & 1 \leq y \leq 2^n - a \\ a+y+1 & 2^n - a < y \leq 2^n \end{cases} \\ &= \sum_{i=1}^{2^n-a} (a+i)(-i) \prod_{\substack{j \neq i \\ 1 \leq j \leq 2^n}} (y-j) + \sum_{i=2^n-a+1}^{2^n} (a+i+1)(-i) \prod_{\substack{j \neq i \\ 1 \leq j \leq 2^n}} (y-j) \\ &= \sum_{i=1}^{2^n} (a+i)(-i) \prod_{\substack{j \neq i \\ 1 \leq j \leq 2^n}} (y-j) + \sum_{i=2^n-a+1}^{2^n} (-i) \prod_{\substack{j \neq i \\ 1 \leq j \leq 2^n}} (y-j). \end{aligned}$$

That is, the function $f(a,y)$ is a polynomial in y with degree at most $2^n - 1$. Moreover, the coefficient of y^{2^n-1} in $f(a,y)$ is

$$\begin{aligned} &\sum_{i=1}^{2^n} (a+i)(-i) + \sum_{i=2^n-a+1}^{2^n} (-i) = -a \sum_{i=1}^{2^n} i - \sum_{i=1}^{2^n} i^2 + \sum_{i=-a}^{-1} (-i) \\ = &\sum_{i=-a}^{-1} (-i) = \sum_{i=1}^{a} i = \frac{a(a+1)}{2}, \end{aligned}$$

which is zero if and only if $a = 0$ or $a = -1 = 2^n$, which cases are excluded by hypothesis. (We have used the facts that $\sum_{i=1}^{2^n} i = 0 \bmod (2^n+1)$, that $\sum_{i=1}^{2^n} i^2 = \frac{1}{6} 2^n (2^n+1)(2 \times 2^n + 1)$, that $2 | 2^n$, and that $3 | 2 \times 2^n + 1$ for $n \in \{2,4,8,16\}$ so that $\sum_{i=1}^{2^n} i^2 = 0 \bmod (2^n+1)$.) Thus, we have shown that the degree of the polynomial $f(a,y)$ is indeed $2^n - 1$.

Note that $f(x,y) = f(y,x)$ for all x and y in $\mathbb{Z}_{2^n+1}$ so that, for every $a \notin \{0, 2^n\}$, $f(x,a)$ is a polynomial in x of degree $2^n - 1$. □

Theorem 4 *If $n \in \{2,4,8,16\}$, then, for every a in $\mathbb{Z}_{2^n} - \{0,1\}$, the function $g(a,x) = a \bigodot x = x \bigodot a$* **cannot** *be written as a polynomial in x over the ring $\mathbb{Z}_{2^n}$.*

We show first the following lemma:

Lemma 1 *If $p(x)$ is a polynomial over $\mathbb{Z}_{2^n}$, then, for all β in $\mathbb{Z}_{2^n}$,*

$$p(2\beta) \bmod 2 = p(0) \bmod 2.$$

Proof. Let $p(x) = a_k x^k + a_{k-1}x^{k-1} + \cdots + a_1 x + a_0$.
Then for all β in $\mathbf{Z}_{2^n}$,

$$p(2\beta) = a_k(2\beta)^k + a_{k-1}(2\beta)^{k-1} + \cdots + a_1 2\beta + a_0.$$

Thus, $p(2x) \bmod 2 = a_0 \bmod 2 = f(0) \bmod 2$. □

Proof of Theorem 4 Let $n > 1$, then for every integer a, $1 < a < 2^n$, there exists an integer $x_0 \in \{1, 2, \ldots, 2^n\}$ such that the following three inequalities all satisfied:

$$2^n + 1 < 2ax_0 < 2(2^n + 1), \tag{3.11}$$

$$0 \leq 2a(x_0 - 1) < 2^n + 1 \tag{3.12}$$

and

$$0 \leq 2x_0 \leq 2^n. \tag{3.13}$$

Inequality (3.11) is equivalent to the inequality $0 < 2ax_0 - (2^n + 1) < 2^n + 1$ with the condition that $2ax_0 - (2^n + 1)$ is an odd integer. Because of (3.13) and from the definition of the function g,

$$g(a, 2x_0) = a \odot (2x_0) = 2ax_0 - (2^n + 1).$$

Thus,

$$g(a, 2x_0) \bmod 2 = (2ax_0 - (2^n + 1)) \bmod 2 = 1.$$

On the other hand, inequality (3.12) implies that $2a(x_0 - 1)$ is an even integer in $\{0, 1, .., 2^n\}$ so that

$$g(a, (2(x_0 - 1)) \bmod 2 = 2a(x_0 - 1) \bmod 2 = 0.$$

Hence, it follows from Lemma 1 that $g(a, x) = a \odot x$ is not a polynomial over $\mathbf{Z}_{2^n}$.
□

3.3 Security Features of IDEA

In this section, we state some provable security features of the IDEA cipher. The security of the IDEA cipher against differential cryptanalysis will be discussed in detail in Chapter 5.

3.3.1 Confusion

The confusion (see page 12) required for a secure cipher is achieved in the IDEA cipher by mixing three incompatible group operations. In the computational graph of the encryption process for IDEA, the three different group operations are so arranged that *the output of an operation of one type is never used as the input to an operation of the same type.*

The three operations are *incompatible* in the sense that:

1. No pair of the 3 operations satisfies a "distributive" law. For instance, for the operations $\odot$ and $\boxplus$, there exist a, b, and c in $\mathbb{F}_2^{16}$, such that,

$$a\boxplus(b\odot c) \neq (a\boxplus b)\odot(a\boxplus c).$$

For example, when $a = b = c = 1 = (0,0,..,0,1)$, the left side of the above inequality is $2 = (0,0,..,0,1,0)$, while the right side equals $4 = (0,0,..,0,1,0,0)$.

2. No pair of the 3 operations satisfies a "generalized associative" law. For instance, for the operations $\boxplus$ and $\bigoplus$, there exist a, b, and c in $\mathbb{F}_2^{16}$, such that,

$$a\boxplus(b\bigoplus c) \neq (a\boxplus b)\bigoplus c.$$

For example, for $a = b = c = 1 = (0,0,...0,1)$ in $\mathbb{F}_2^{16}$, the left side of the above inequality is $1 = (0,0,...0,1)$, while the right side equals $3 = (0,0,..,0,1,1)$. Thus, one cannot arbitrarily change the order of operations to simplify analysis.

3. The 3 operations are connected by the direct mapping d and its inverse, which inhibits isotopisms as was shown in Theorem 2. The cryptographic significance of this fact is that, if there were an isotopism between two operations, then one could replace one operation with the other by applying bijective mappings on the inputs and on the output. It follows from Theorem 2 that $(\mathbb{F}_2^{16},\odot)$ and $(\mathbb{F}_2^{16},\bigoplus)$ are not isotopic and that $(\mathbb{F}_2^{16},\boxplus)$ and $(\mathbb{F}_2^{16},\bigoplus)$ are not isotopic. The isotopism from $(\mathbb{F}_2^{16},\odot)$ onto $(\mathbb{F}_2^{16},\boxplus)$ is essentially the discrete logarithm, which , as shown in Theorem 2, cannot be the direct mapping d. Moreover, the discrete logarithm is generally considered to be a "complex" function.

4. Under the direct mapping d and its inverse d^{-1}, it is possible to consider the operations $\odot$ and $\boxplus$ as acting on the same set (either in the ring $\mathbb{Z}_{2^n}$ or in the field $\mathbb{Z}_{2^n+1}$). However, by doing so, we must analyze some highly non-linear functions in

the sense that multiplication modulo $2^{16}+1$, which is a bilinear function over $\mathbb{Z}_{2^{16}+1}$, corresponds to a non-polynomial function over $\mathbb{Z}_{2^{16}}$, as was shown in Theorem 4. Similarly, addition modulo 2^{16}, which is an affine function in each argument over $\mathbb{Z}_{2^{16}}$, corresponds to a two variable polynomial of degree $2^{16}-1$ in each variable over $\mathbb{Z}_{2^{16}+1}$, as was shown in Theorem 3. [Note that every function h from $\mathbb{Z}_{2^{16}+1}$ to $\mathbb{Z}_{2^{16}+1}$ is a polynomial of degree at most 2^{16}. Moreover, if such a function is invertible then its degree is at most $2^{16}-1$ as follows from (3.10) and from the facts that function $h(x)$ is invertible if and only if function $h(x)-h(0)$ is invertible and that these two functions have the same degree].

3.3.2 Diffusion

A check by direct computation has shown that the round function is "complete", i.e., that each output bit of the first round depends on every bit of the plaintext and on every bit of the key used for that round. This diffusion is provided in the IDEA cipher by the transformation called the multiplication-addition (MA) structure whose computational graph is shown in Fig.3.2. The MA structure transforms two 16 bit subblocks into two 16 bit subblocks controlled by two 16 bit key subblocks. This structure has the following properties:

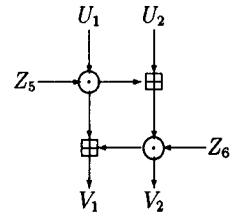


Figure 3.2: Computational graph of the MA structure.

- for any choice of the key subblocks Z_5 and Z_6, $MA(\cdot,\cdot,Z_5,Z_6)$ is an invertible transformation; for any choice of U_1 and U_2, $MA(U_1,U_2,\cdot,\cdot)$ is also an invertible transformation;

- this structure has a "complete diffusion" effect in the sense that each output subblock depends on every input subblock, and

– this structure uses the least number of operations (four) required to achieve such complete diffusion. [To give a formal proof of this property, we need the following definitions.

An *operation* is a mapping from two variables to one variable. A *computational graph* of a function is a directed graph in which the vertices are operations, the edges entering a vertex are the inputs to the operation, the edges leaving a vertex are the output variable of the operation, the edges entering no vertex are the output variables, and the edges leaving no vertex are the inputs variables. An algorithm to compute a function determines a computational graph where the input variables are the input to the algorithm and the output variables are the outputs of the algorithm.

Consider a function having the form

$$(Y_1, Y_2) = E(X_1, X_2, Z_1, Z_2), \qquad X_i, Y_i \in \mathbb{F}_2^n, \quad Z_i \in \mathbb{F}_2^k \tag{3.14}$$

and such that, for every choice of (Z_1, Z_2), $E(\cdot, \cdot, Z_1, Z_2)$ is invertible. Such a function will be called a *2-block cipher*. A 2-block cipher will be said to have *complete diffusion* if each of its output variable depends non-idly on every input variable.

Lemma 2 *If a 2-block cipher of the form (3.14) has complete diffusion, then the computational graph determined by any algorithm that computes the cipher function contains at least 4 operations.*

Proof. Let $Y_1 = E_1(X_1, X_2, Z_1, Z_2)$, and $Y_2 = E_2(X_1, X_2, Z_1, Z_2)$. Because E_1 has complete diffusion, its computational graph contains at least 3 operations because this function has four input variables. Suppose E_1 contains exactly 3 operations. The invertibility of the 2-block cipher implies that $E_2 \neq E_1$ and complete diffusion requires that E_2 not equal any intermediate result that appears in E_1. Thus, at least one operation not appearing in E_1 is required in the computational graph of E_2. This proves the lemma.]

3.3.3 Perfect secrecy for a "one-time" key

Perfect secrecy (see page 7) in the sense of Shannon is obtained in each round of encryption if a "one-time" key (see page 7) is used. In fact, such perfect secrecy is achieved at the input transformation in the first round because each operation is a group operation. In addition, for every choice of (p_1, p_2, p_3, p_4) and of (q_1, q_2, q_3, q_4)

in $\mathbb{F}_2^{64}$, there are exactly 2^{32} different choices of the key subblocks $(Z_1, .., Z_6)$ such that the first round of the cipher transforms (p_1, p_2, p_3, p_4) into (q_1, q_2, q_3, q_4).

3.4 Implementations of the Cipher

The cipher IDEA can be easily implemented in software because only basic operations on pairs of 16-bit subblocks are used in the encryption process. A C-language program implementing the cipher and some sample data for checking the correctness of implementation are given in Section 3.4.3. This C-program can achieve data-rates from about 200 Kbits per second on an IBM-PC to about 3.2 Mbits per second on a VAX-9000.

The regular modular structure of the cipher facilitates hardware implementations. The similarity of encryption and decryption for the IDEA cipher, shown in next section, makes it possible to use the same device in both encryption and decryption. An algorithm for computing the operation $\odot$ is described in Section 3.4.2.

3.4.1 Similarity of encryption and decryption

The *similarity* of encryption and decryption means that decryption is essentially the same process as encryption, the only difference being that different key subblocks are used. Thus, the same device can be used for both encryption and decryption, the only "extra" cost being the pre-computation of the key subblocks from the 128-bit secret key. In the following we show that the round function of the IDEA cipher has the form (2.2) on page 17, that is, the round function consists of a group cipher followed by an involution cipher plus an involutary permutation which is an automorphism of the group $(\mathbb{F}_2^{64}, \otimes)$. Then it follows from Theorem 1 (see page 17) that IDEA cipher has similarity of encryption and decryption.

For the encryption process of the IDEA cipher shown in Fig.3.1, define

$$X \otimes Z_A = (X_1 \odot Z_1, X_2 \boxplus Z_2, X_3 \boxplus Z_3, X_4 \odot Z_4),$$

then it is easy to see that $(\mathbb{F}_2^{64}, \otimes)$ is a group.

Let $P_I(X)$ be the permutation on X that interchanges the subblocks X_2 and X_3 of $X = (X_1, X_2, X_3, X_4)$ at the end of each round. It is obvious that P_I is an involution and that $P_I(X \otimes Z_A) = P_I(X) \otimes P_I(Z_A)$, so that P_I is an automorphism of the group $(\mathbb{F}_2^{64}, \otimes)$.

It remains to show that the function $In(\cdot, Z_B)$, shown in Fig.3.3, with the 64-bit input (S_1, S_2, S_3, S_4) and the 64-bit output (T_1, T_2, T_3, T_4) controlled by the 32-bit

key $Z_B = (Z_5, Z_6)$, is an involution. That is, for any fixed Z_B, the inverse of the function $In(\cdot, Z_B)$ is itself. This self-inverse property is a consequence of the fact that the exclusive-OR of (S_1, S_2) and (S_3, S_4) is equal to the exclusive-OR of (T_1, T_2) and (T_3, T_4); Thus, the input to the MA structure in Fig. 3.2 is unchanged when S_1, S_2, S_3 and S_4 are replaced by T_1, T_2, T_3 and T_4.

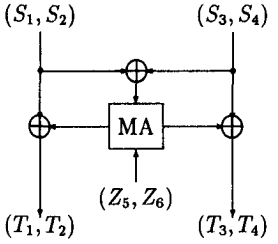


Figure 3.3: Computational graph of the involution $In(\cdot, Z_B)$.

3.4.2 Low-High algorithm for multiplication

The most difficult step in the implementation of the IDEA cipher is the implementation of multiplication modulo $(2^{16} + 1)$, which can be implemented in the way suggested by the following lemma.

Lemma 3 [Low-High algorithm for $\odot$] *Let a, b be two n-bit non-zero integers in $\mathbb{Z}_{2^n+1}$, then*

$$ab \bmod (2^n + 1) = \begin{cases} (ab \bmod 2^n) - (ab \operatorname{div} 2^n) & \text{if } (ab \bmod 2^n) \geq (ab \operatorname{div} 2^n) \\ (ab \bmod 2^n) - (ab \operatorname{div} 2^n) + 2^n + 1 & \text{if } (ab \bmod 2^n) < (ab \operatorname{div} 2^n) \end{cases}$$

where $(ab \operatorname{div} 2^n)$ denotes the quotient when ab is divided by 2^n.

Note that $(ab \bmod 2^n)$ corresponds to the n least significant bits of ab, and $(ab \operatorname{div} 2^n)$ is just the right-shift of ab by n bits. Note also that $(ab \bmod 2^n) = (ab \operatorname{div} 2^n)$ implies that $ab \bmod (2^n + 1) = 0$ and hence cannot occur when $2^n + 1$ is a prime.

Proof. For any non-zero a and b in $\mathbb{Z}_{2^n+1}$, there exist unique integers q and r such that

$$ab = q(2^n + 1) + r, \qquad 0 \leq r < 2^n + 1,\ 0 \leq q < 2^n.$$

Moreover, $q + r < 2^{n+1}$. Note that $r = ab \bmod (2^n + 1)$. We have

$$(ab \text{ div } 2^n) = \begin{cases} q & \text{if } q + r < 2^n \\ q + 1 & \text{if } q + r \geq 2^n \end{cases}$$

and

$$(ab \bmod 2^n) = \begin{cases} q + r & \text{if } q + r < 2^n \\ q + r - 2^n & \text{if } q + r \geq 2^n. \end{cases}$$

Thus

$$r = \begin{cases} (ab \bmod 2^n) - (ab \text{ div } 2^n) & \text{if } q + r < 2^n \\ (ab \bmod 2^n) - (ab \text{ div } 2^n) + 2^n + 1 & \text{if } q + r \geq 2^n. \end{cases}$$

But $q + r < 2^n$ if and only if $(ab \bmod 2^n) \geq (ab \text{ div } 2^n)$. This proves the Lemma. □

Remark. There are of course other ways to compute the operation $\odot$. For example, based on the fact that

$$x \cdot y = \alpha^{(\log_\alpha(x) + \log_\alpha(y) \bmod 2^n)} \bmod (2^n + 1)$$

for all x and y in $\mathbb{Z}^*_{2^n+1}$ where α is a generator of the cyclic group $\mathbb{Z}^*_{2^n+1}$, one can compute $\odot$ by using $\boxplus$ together with look-up tables for computing $\log_\alpha(\cdot)$ and $\alpha^{(\cdot)}$. For small n, i.e., for $n = 2, 4$ or 8, this is more efficient than the Low-High algorithm. However, for $n = 16$, this method requires more memory. More details can be found in [10].

3.4.3 C-program of IDEA cipher and sample data

```
/* C - program  of block cipher IDEA */

#include <stdio.h>
# define maxim 65537
# define fuyi  65536
# define one   65535
# define round  8
void   cip(unsigned IN[5],unsigned OUT[5],unsigned Z[7][10]);
void   key( short unsigned uskey[9],unsigned Z[7][10] );
void   de_key(unsigned Z[7][10],unsigned DK[7][10]);
unsigned inv(unsigned xin);
unsigned mul(unsigned a, unsigned b);

main()
{
  int       i, j, k, x;
  unsigned  Z[7][10], DK[7][10], XX[5],TT[5], YY[5];
  short unsigned uskey[9];
   for( i=1; i<=8; i++ ) uskey[i]= i;
   key(uskey,Z);                    /* generate encryption subkeys Z[i][r] */
```

```
   printf("\n encryption keys    Z1     Z2     Z3     Z4     Z5     Z6");
   for( j=1; j<=9; j++ ) {    printf("\n %3d-th round  ", j);
         if (j==9)  for( i=1; i<=4; i++ )     printf(" %6d",Z[i][j]);
             else   for( i=1; i<=6; i++ )     printf(" %6d",Z[i][j]);
     }
   de_key(Z,DK);                    /* compute decryption subkeys DK[i][r] */

   printf("\n \n decryption keys   DK1    DK2    DK3    DK4    DK5    DK6 ");
   for( j=1; j<=9; j++ ) {    printf("\n %3d-th round  ", j);
         if (j==9)  for( i=1; i<=4; i++ )     printf(" %6d",DK[i][j]);
             else   for( i=1; i<=6; i++ )     printf(" %6d",DK[i][j]);
        }
        for (x=1; x<=4; x++) XX[x]=x-1;
        printf("\n \n plaintext X  %6u  %6u  %6u  %6u  \n",
                                 XX[1], XX[2], XX[3], XX[4]);

        cip(XX,YY,Z);               /* encipher XX to YY with key Z   */

        printf("\n \n ciphertext Y  %6u  %6u  %6u  %6u  \n",
                                 YY[1], YY[2], YY[3], YY[4]);

        cip(YY,TT,DK);              /* decipher YY to TT with key DK */

        printf("\n \n result     T  %6u  %6u %6u  %6u  \n",
                                 TT[1], TT[2], TT[3], TT[4]);
   }

    /* encryption algorithm  */
void  cip(unsigned IN[5],unsigned OUT[5],unsigned Z[7][10])
 {
    unsigned int r, x1,x2,x3,x4,kk,t1,t2,a;
     x1=IN[1];  x2=IN[2]; x3= IN[3]; x4=IN[4];
     for (r= 1; r<= 8; r++)                      /* the round function     */
      {
            /* the group operation on 64-bits block */
       x1 =mul(x1,Z[1][r]);          x4 =mul(x4,Z[4][r]);
       x2 =( x2 + Z[2][r] ) & one;   x3 =( x3 + Z[3][r] ) & one;
            /* the function of the MA structure */
       kk = mul( Z[5][r], ( x1^x3 ) );
       t1 = mul( Z[6][r], ( kk + ( x2^x4 ) )& one);
       t2 = ( kk + t1 ) & one;
            /*  the involutary permutation PI  */
       x1 = x1^t1;        x4 = x4^t2;
       a = x2^t2;         x2 = x3^t1;       x3 = a;
       printf("\n     %1u-th rnd  %6u  %6u  %6u  %6u  ", r, x1, x2, x3, x4);
      }
            /*   the output transformation      */
    OUT[1] = mul( x1,Z[1][round+1] );
    OUT[4] = mul( x4,Z[4][round+1] );
    OUT[2] =  ( x3 + Z[2][round +1] )& one;
    OUT[3] =  ( x2 + Z[3][round+1] ) & one;
 }
```

```
   /*  multiplication using the Low-High algorithm  */
unsigned mul(unsigned a, unsigned b)
{
  long int p;
  long unsigned q;
     if (a==0)    p = maxim-b;
     else if ( b==0 )  p = maxim-a;  else
       { q=(unsigned long)a*(unsigned long)b;
         p=( q & one) - (q>>16);  if (p<=0) p= p+maxim;
       }
  return (unsigned)(p & one);
}


        /*  compute inverse of xin by Euclidean gcd alg. */
unsigned inv(unsigned  xin)
{   long n1,n2,q,r,b1,b2,t;
     if ( xin == 0 )  b2 = 0;
     else
      {  n1=maxim; n2 = xin; b2= 1; b1= 0;
         do   {  r = (n1 % n2);  q = (n1-r)/n2 ;
                 if (r== 0)  {if ( b2<0 )  b2 = maxim+b2; }
                 else   { n1= n2; n2= r; t = b2; b2= b1- q*b2; b1= t; }
              } while (r != 0);
      }
   return (unsigned)b2;
 }


     /*  generate encryption  subkeys Z's   */
void   key( short unsigned uskey[9], unsigned  Z[7][10] )
{
  short unsigned S[54];
  int  i,j,r;
 for (i = 1; i<9; i++)  S[i-1] = uskey[i];
   /*   shifts   */
 for (i = 8; i< 54; i++ )
  {
     if ( (i+2)%8 == 0 )                        /* for S[14],S[22],..  */
        S[i] = ( ( S[i-7] <<9 )^( S[i-14] >>7 ) ) & one;
     else  if ( (i+1)%8 ==0 )                   /* for S[15],S[23],..   */
        S[i] =( ( S[i-15] <<9 )^( S[i-14] >>7 ) ) & one ;
     else
        S[i] = ( ( S[i-7] <<9 )^( S[i-6] >>7 )) & one;
   }
    /* get subkeys     */
      for (r= 1; r<=round+1; r++)   for(j= 1;j<7; j++)
                Z[j][r] = S[6*(r-1) + j-1];
 }
```

```
    /* compute decryption subkeys DK's  */
void  de_key(unsigned Z[7][10],unsigned DK[7][10])
{
   int  j;
   for (j = 1; j<=round+1; j++)
    {   DK[1][round-j+2] = inv(Z[1][j]);
        DK[4][round-j+2] = inv(Z[4][j]);
     if ( j==1 || j==round+1 ){
        DK[2][round-j+2] = ( fuyi-Z[2][j] ) & one;
        DK[3][round-j+2] = ( fuyi-Z[3][j] ) & one;
        }
     else {
        DK[2][round-j+2] = ( fuyi-Z[3][j] ) & one;
        DK[3][round-j+2] = ( fuyi-Z[2][j] ) & one;
        }
     }
   for (j= 1;j<=round+1;j++)
      { DK[5][round+1-j] = Z[5][j];  DK[6][round+1-j] = Z[6][j];}
 }
```

Sample Data. All the numbers are 16-bit integers with the leftmost bit being the most significant bit.

encryption keys	Z1	Z2	Z3	Z4	Z5	Z6
1-th round	1	2	3	4	5	6
2-th round	7	8	1024	1536	2048	2560
3-th round	3072	3584	4096	512	16	20
4-th round	24	28	32	4	8	12
5-th round	10240	12288	14336	16384	2048	4096
6-th round	6144	8192	112	128	16	32
7-th round	48	64	80	96	0	8192
8-th round	16384	24576	32768	40960	49152	57345
9-th round	128	192	256	320		

decryption keys	DK1	DK2	DK3	DK4	DK5	DK6
1-th round	65025	65344	65280	26010	49152	57345
2-th round	65533	32768	40960	52428	0	8192
3-th round	42326	65456	65472	21163	16	32
4-th round	21835	65424	57344	65025	2048	4096
5-th round	13101	51200	53248	65533	8	12
6-th round	19115	65504	65508	49153	16	20
7-th round	43670	61440	61952	65409	2048	2560
8-th round	18725	64512	65528	21803	5	6
9-th round	1	65534	65533	49153		

plaintext X	0	1	2	3
after				
1-th rnd	240	245	266	261
2-th rnd	8751	8629	62558	59737
3-th rnd	3974	14782	36584	4467
4-th rnd	22495	44120	50779	47693
5-th rnd	36481	47772	63359	14922
6-th rnd	26946	37897	57883	7268
7-th rnd	39376	51190	21297	25102
8-th rnd	2596	152	60523	18725
ciphertext Y	4603	60715	408	28133
after				
1-th rnd	55693	54065	10230	33464
2-th rnd	48205	57963	37961	42358
3-th rnd	2724	63471	55964	9443
4-th rnd	51782	65115	56408	4461
5-th rnd	29839	36616	14810	17868
6-th rnd	12902	1118	12213	45102
7-th rnd	1680	1290	253	7674
8-th rnd	0	5	3	12
result T	0	1	2	3

Chapter 4

Markov Ciphers and Differential Cryptanalysis

In this chapter we consider the security of iterated block ciphers against the differential cryptanalysis introduced by Biham and Shamir [6]. Differential cryptanalysis is a chosen-plaintext attack on iterated ciphers. It analyzes the effect of the "difference" of a pair of plaintexts on the "difference" of succeeding round outputs in an r-round iterated cipher. In section 4.1, the concept of "Markov ciphers" is introduced because of its significance in differential cryptanalysis. In Section 4.2, differential cryptanalysis of a general r-round iterated cipher is described in terms of $(r-1)$-round "differentials", instead of in terms of the "i-round characteristics" used in [6]. The hypothesis of stochastic equivalence, which has been implicitly assumed in differential cryptanalysis, is explicitly formulated. Section 4.3 considers the (data) complexity of differential cryptanalysis attack. It is shown that the probabilities of such differentials can be used to determine a lower bound on the complexity of a differential cryptanalysis attack and to show when an r-round cipher is not vulnerable to such attacks. The security of Markov ciphers against differential cryptanalysis attacks is considered in Section 4.4. It is shown that for a "good" Markov cipher, the complexity of differential cryptanalysis attack will increase exponentially with the number of rounds. A design principle for Markov ciphers, that their transition matrices should be non-symmetric, is established.

4.1 Markov Ciphers

Throughout this chapter, we consider the encryption of a pair of *distinct* plaintexts by an r-round iterated cipher as shown schematically in Fig.4.1. In this figure, the

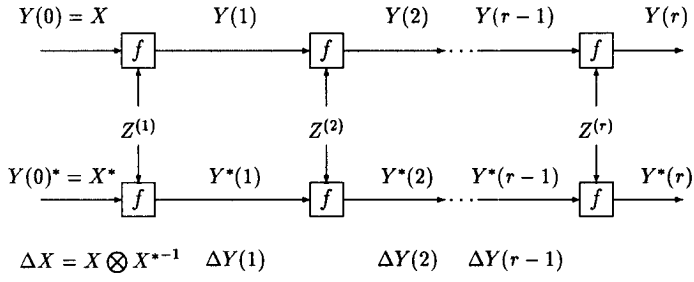


Figure 4.1: The sequence of differences–Encrypting a pair of plaintexts with an r-round iterated cipher.

difference $\Delta Y(i)$ between two m-bit blocks $Y(i)$ and $Y^*(i)$ is defined as

$$\Delta Y(i) = Y(i) \otimes Y^*(i)^{-1},$$

where $\otimes$ denotes a specified group operation on the set of m-bit blocks and $Y^*(i)^{-1}$ denotes the inverse of the element $Y^*(i)$ in the group.

From the pair of encryptions, one obtains the

$$\textit{sequence of differences} \qquad \Delta Y(0), \Delta Y(1), ..., \Delta Y(r)$$

where $Y(0) = X$ and $Y^*(0) = X^*$ denote the plaintext pair so that $\Delta Y(0) = \Delta X$, and where $Y(i)$ and $Y^*(i)$ for $(0 < i < r)$ are the outputs of the i-th round, which are also the inputs to the (i+1)-th round. The subkey for the i-th round is denoted as $Z^{(i)}$ and f is the round function such that $Y(i) = f(Y(i-1), Z^{(i)})$.

In the following discussion, we always assume that $X \neq X^*$ because, when $X = X^*$, all $\Delta Y(i)$ would equal the *neutral* element e of the group, which case is of no interest for differential cryptanalysis. Thus, $\Delta Y(i) \in \mathbb{F}_2^m - \{e\}$. We assume also that the subkeys used in each round of the iterated cipher are statistically independent and uniformly distributed, but the results apply well in the practice to iterated ciphers whose round subkeys are generated by a key-schedule algorithm from the secret key.

Recall that a sequence of discrete random variables $v_0, v_1, ..., v_r$ is a *Markov chain* if, for $0 \leq i < r$ (where $r = \infty$ is allowed),

$$P(v_{i+1} = \beta_{i+1} | v_i = \beta_i, v_{i-1} = \beta_{i-1}, ..., v_0 = \beta_0) = P(v_{i+1} = \beta_{i+1} | v_i = \beta_i).$$

A Markov chain is called *homogeneous* if $P(v_{i+1} = \beta | v_i = \alpha)$ is independent of i for all α and β.

Definition 3 *An iterated cipher with round function f is a* Markov cipher *if there is a group operation $\bigotimes$ for defining differences such that, for all choices of α ($\alpha \neq e$) and β ($\beta \neq e$),*

$$P(\Delta Y = \beta | \Delta X = \alpha, X = \gamma),$$

where $Y = f(X, Z)$ and $Y^* = f(X^*, Z)$, is independent of γ when the subkey Z is uniformly random, or, equivalently, if

$$P(\Delta Y = \beta | \Delta X = \alpha, X = \gamma) = P(\Delta Y(1) = \beta_1 | \Delta X = \alpha)$$

for all choices of γ when the subkey Z is uniformly random.

The following crucial theorem explains the terminology "Markov cipher".

Theorem 5 *If an r-round iterated cipher is a Markov cipher and the r round subkeys are independent and uniformly random, then the sequence of differences $\Delta X = \Delta Y(0), \Delta Y(1), ..., \Delta Y(r)$ is a homogeneous Markov chain. Moreover, this Markov chain is stationary if ΔX is uniformly distributed over the non-neutral elements of the group.*

Proof. To show that the sequence $\Delta X, \Delta Y(1), ..., \Delta Y(r)$ is a Markov chain, it is sufficient to show for the second round that

$$P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1, \Delta X = \alpha) = P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1).$$

To show this, we note that

$$\begin{aligned}
&P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1, \Delta X = \alpha) \\
&= \sum_{\gamma} P(Y(1) = \gamma, \Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1, \Delta X = \alpha) \\
&= \sum_{\gamma} P(Y(1) = \gamma | \Delta Y(1) = \beta_1, \Delta X = \alpha) \\
&\quad \times P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1, Y(1) = \gamma, \Delta X = \alpha) \\
&= \sum_{\gamma} P(Y(1) = \gamma | \Delta Y(1) = \beta_1, \Delta X = \alpha) P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1, Y(1) = \gamma) \\
&= \sum_{\gamma} P(Y(1) = \gamma | \Delta Y(1) = \beta_1, \Delta X = \alpha) P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1) \\
&= P(\Delta Y(2) = \beta_2 | \Delta Y(1) = \beta_1),
\end{aligned}$$

where the third equality comes from the fact that $Y(1)$ and $\Delta Y(1)$ together determine both $Y(1)$ and $Y(1)^*$ so that $\Delta Y(2)$ has no further dependence on ΔX when

$Y(1)$ and $\Delta Y(1)$ are specified. Because the same round function is used in each round, this Markov chain is homogeneous. For any key $Z = z$, the round function $f(\cdot, z)$ is a bijective mapping from the set of plaintexts to the set of ciphertexts. This bijection induces a bijection from pairs of distinct plaintexts (X, X^*) to pairs of distinct ciphertexts $(Y, Y^*) = (f(X, z), f(X^*, z))$. The fact that X and $\Delta X(\neq e)$ are independent and uniformly distributed implies that (X, X^*) is uniformly distributed over pairs of distinct plaintexts. Thus, (Y, Y^*) is also uniformly distributed over pairs of distinct ciphertexts and hence $\Delta Y(\neq e)$ is also uniformly distributed. Thus the uniform distribution is a stationary distribution for this Markov chain. □

Example 4
1) Block cipher DES is a Markov cipher under the definition of difference as $\Delta X = X \bigoplus (X^*)^{-1} = X \bigoplus X^*$ *where* $\bigoplus$ *denotes bitwise XOR.*
2) For the definition of difference $\Delta X = X \bigoplus X^*$, *one can show that the block ciphers LOKI [11], FEAL [54] (both are DES-like iterated ciphers) and REDOC [13] are also Markov ciphers.*

In the above example, 1) is just a restatement of Lemma 1 in [6]. The second part 2) can be shown by using the similar argument as in the proof of the following theorem.

Theorem 6 *If the round function of an iterated cipher has the form*

$$f(X, Z) = g(X \otimes Z_A, Z_B)$$

where $\otimes$ *is a group operation for* $\mathbb{F}_2^m$ *and where the function* $g(\cdot, Z_B)$ *is invertible for every choice of* Z_B, *then the iterated cipher is a Markov cipher under the definition of difference as* $\Delta X = X \otimes (X^*)^{-1}$.

Remark. The iterated ciphers of constructions III and IV in Section 2.4.2 are of this form. Thus, the block cipher IDEA is a Markov cipher as also is its predecessor PES.

Proof. Let $S = X \otimes Z_A$ and $Y = g(S, Z_B)$, then $S^* = X^* \otimes Z_A$ and $Y^* = g(S^*, Z_B)$. Thus,

$$\Delta S = S \otimes (S^*)^{-1} = (X \otimes Z_A) \otimes (Z_A^{-1} \otimes (X^*)^{-1}) = \Delta X.$$

Because

$$\Delta Y = g(S, Z_B) \otimes (g(S^*, Z_B))^{-1} = g(S, Z_B) \otimes (g((\Delta S)^{-1} \otimes S, Z_B))^{-1},$$

it follows that ΔY has no further dependence on X when ΔS and S are specified. Thus,

$$\begin{aligned}
&P(\Delta Y = \beta | \Delta X = \beta, X = \gamma) \\
&= P(\Delta Y = \beta | \Delta S = \beta, X = \gamma) \\
&= \sum_{\lambda} P(\Delta Y = \beta, S = \lambda | \Delta S = \beta, X = \gamma) \\
&= \sum_{\lambda} P(\Delta Y = \beta | \Delta S = \beta, X = \gamma, S = \lambda) P(S = \lambda | \Delta S = \beta, X = \gamma) \\
&= \sum_{\lambda} P(\Delta Y = \beta | \Delta S = \beta, S = \lambda) P(Z_A = \lambda \otimes \gamma^{-1}) \\
&= 2^{-m} \sum_{\lambda} P(\Delta Y = \beta | \Delta S = \beta, S = \lambda),
\end{aligned}$$

which is independent of γ, where we have used the facts that

$$P(S = \lambda | \Delta S = \beta, X = \gamma) = P(S = \lambda | X = \gamma) = P(Z_A = \lambda \otimes \gamma^{-1}).$$

and that the subkey $Z = (Z_A, Z_B)$ is uniformly random. □

For any Markov cipher, let Π denote the *transition probability matrix* of the homogeneous Markov chain $\Delta X = \Delta Y(0), \Delta Y(1), ..., \Delta Y(r)$. The (i, j) entry in Π is $P(\Delta Y(1) = \alpha_j | \Delta X = \alpha_i)$ where $\alpha_1, \alpha_2, ..., \alpha_M$ is some agreed-upon ordering of the M possible values of ΔX and $M = 2^m - 1$ for an m-bit cipher. Then, for every $r \geq 1$,

$$\Pi^r = \left[p_{ij}^{(r)}\right] = \left[P(\Delta Y(r) = \alpha_j | \Delta X = \alpha_i)\right]. \tag{4.1}$$

Note that each row of Π is a probability distribution so that the sum of the elements in each row of Π is 1. From Theorem 5, the uniform distribution is a stationary distribution. Thus, for each j, we have

$$\frac{1}{2^m - 1} = \sum_{i=1}^{2^m - 1} p_{ij} \frac{1}{2^m - 1} = \frac{1}{2^m - 1} \sum_{i=1}^{2^m - 1} p_{ij}$$

so that every column of Π also sums to 1. Thus, we have the following result.

Theorem 7 *The transition matrix of a Markov cipher is doubly stochastic, i.e., every row sum is 1 and every column sum is 1.*

4.2 Differential Cryptanalysis

4.2.1 The round differentials

Differential cryptanalysis, proposed by Biham and Shamir in [6], is a chosen-plaintext attack on iterated ciphers. It is probably the best method presently known for attacking iterated ciphers [6, 7, 8, 9, 32]. An iterated cipher is based on iterating a "simple" round function several times. Differential cryptanalysis exploits the fact that the simple round function f in an iterated cipher is usually *cryptographically weak* in the sense that for $Y(i) = f(Y(i-1), Z^{(i)})$ and $Y^*(i) = f(Y^*(i-1), Z^{(i)})$, if one or more values of the triple $(\Delta Y(i-1), Y(i), Y^*(i))$ are known, then it is feasible to determine the round subkey $Z^{(i)}$. Thus, if the ciphertext pair is known and the difference of the pair of inputs to the last round can somehow be obtained, then it is usually feasible to determine (some substantial part of) the subkey of the last round. In differential cryptanalysis, this is achieved by *choosing* plaintext pairs (X, X^*) with a specified difference α such that the difference $\Delta Y(r-1)$ of the pair of inputs to the last round will take on a particular value β with high probability. Based on this idea, we make the following definition.

Definition 4 *An i-round differential is a couple (α, β), where α is the difference of a pair of distinct plaintexts X and X^* and where β is a possible difference for the resulting i-th round outputs $Y(i)$ and $Y^*(i)$. The probability of an i-round differential (α, β) is the conditional probability that β is the difference $\Delta Y(i)$ of the ciphertext pair after i rounds given that the plaintext pair (X, X^*) has difference $\Delta X = \alpha$ when the plaintext X and the subkeys $Z^{(1)}, ..., Z^{(i)}$ are independent and uniformly random. We denote this differential probability by $P(\Delta Y(i) = \beta | \Delta X = \alpha)$.*

From (4.1) we see that for a Markov cipher, the probability of the i-round differential is just the (α, β) entry in the i-th transition matrix Π^i.

4.2.2 Differential cryptanalysis attack

The basic procedure of a differential cryptanalysis *attack* on an r-round iterated cipher can be summarized as follows:

1) Find an $(r-1)$-round differential (α, β) such that $P(\Delta Y(r-1) = \beta | \Delta X = \alpha)$ has maximum, or nearly maximum, probability.

2) Choose a plaintext X uniformly at random and compute X^* so that the difference ΔX between X and X^* is α. Submit X and X^* for encryption under the actual

key Z. From the resultant ciphertexts $Y(r)$ and $Y^*(r)$, find every possible value (if any) of the subkey $Z^{(r)}$ of the last round corresponding to the anticipated difference $\Delta Y(r-1) = \beta$. Add one to the count of the number of appearances of each such value of the subkey $Z^{(r)}$.

3) Repeat 2) until one or more values of the subkey $Z^{(r)}$ are counted significantly more often than the others. Take this most-often-counted subkey, or this small set of such subkeys, as the cryptanalyst's decision for the actual subkey $Z^{(r)}$.

Remark. In this chapter only the chosen-plaintext attack by differential cryptanalysis using *one differential* is considered. We remark here that if more highly-probable differentials are known in advance, then differential cryptanalysis can be done more efficiently in the sense described below. Let Ω be the set of plaintext differences known in advance to the attacker such that, for each $\alpha \in \Omega$, there is an $(r-1)$-round differential (α, β_α) with high probability. An attack by differential cryptanalysis based on the knowledge of Ω can be carried out as follows:

For each pair of (known or chosen) plaintext/ciphertext pairs (X,Y) and (X^*,Y^*), if the difference $\Delta X = X \otimes (X^*)^{-1} = \alpha$ belongs to the set Ω, then find every possible value (if any) of the subkey $Z^{(r)}$ of the last round from the two corresponding ciphertexts and the anticipated difference $\Delta Y(r-1) = \beta_\alpha$. Add one to the count of the number of appearances of each such value of the subkey $Z^{(r)}$. Repeat the above step for every pair of such plaintext/ciphertext pairs. If some values of the subkey $Z^{(r)}$ are counted significantly more often than the others, then these values are taken as the possible values for the actual subkey $Z^{(r)}$ and can be checked by other consistency tests to obtain the cryptanalyst's decision for the actual subkey $Z^{(r)}$.

Suppose that the chosen-plaintext attack using one differential needs T pairs of encryptions and suppose that the set Ω contains N elements and the N corresponding differentials have roughly the same probability. Then a chosen-plaintext attack as described above needs only about $\frac{T}{\sqrt{N}}$ chosen plaintexts because one can choose $\sqrt{N}$ plaintexts in such a way that every pair of these plaintexts has a difference belonging to the set Ω so that $\sqrt{N}$ plaintexts produce N differences useful for the attack.

4.2.3 Hypothesis of stochastic equivalence

In a differential cryptanalysis attack, all the subkeys are *fixed*, only the plaintext can be randomly chosen. In the computation of a differential probability, however, the plaintext and all subkeys are independent and uniformly random. In preparing a differential cryptanalysis attack, one uses the computed differential probabilities to determine which differential to use in the attack; hence, one is tacitly making the following hypothesis.

Hypothesis of Stochastic Equivalence. *For virtually all high probability* $(r-1)$*-round differentials* (α, β),

$$\begin{aligned} &P(\Delta Y(r-1) = \beta | \Delta X = \alpha) \\ &\quad \approx P(\Delta Y(r-1) = \beta | \Delta X = \alpha, Z^{(1)} = z_1, ..., Z^{(r-1)} = z_{r-1}) \end{aligned} \tag{4.2}$$

holds for a substantial fraction of the subkey values $(z_1, ..., z_{r-1})$.

A high probability $(r-1)$-round differential will be called *useful in a differential cryptanalysis attack (DC-useful)* if (4.2) holds for this differential.

From the description of a differential cryptanalysis attack and from the fact that there are $2^m - 1$ possible values of $\Delta Y(r-1)$ for a m-bit block cipher, one deduces the following result.

Theorem 8 *Suppose the hypothesis of stochastic equivalence is true, then an* r*-round iterated cipher with independent subkeys is vulnerable to differential cryptanalysis if and only if the round function is weak and there exists a DC-useful* $(r-1)$*-round differential* (α, β) *such that* $P(\Delta Y(r-1) = \beta | \Delta X = \alpha) \gg \frac{1}{2^m - 1}$, *where* m *is the block length of the cipher.*

From the above discussion, it can be seen that the security of an iterated cipher against a differential cryptanalysis attack depends crucially on the differential probabilities, which further depend on the choice of the group operation used for defining "difference". For differential cryptanalysis to be successful, the group operation should be chosen to *maximize the probability of DC-useful differentials.* The choice of group operation that makes the cipher Markov appears to be the most appropriate; this can be seen from Biham and Shamir's work as well as from the analysis in Chapter 5.

4.3 Complexity of Differential Cryptanalysis Attack

From our study on the differential cryptanalysis attack, we see that the *data-complexity* (see page 10) of the attack is twice the number of necessary plaintext pairs (X, X^*) chosen for double encryption. The *processing-complexity* of the differential cryptanalysis attack is essentially the amount of computation used to find the possible values for the subkey $Z^{(r)}$ from the triples $(\Delta Y(r-1), Y, Y^*)$, which is in fact independent of r and is in most cases relatively small [9] because the round function is cryptographically weak. The complexity of differential cryptanalysis attack of an r-round cipher has been defined in [6] as the number of encryptions used. Note that this number is our measure for the data-complexity.

Let $C_d(r)$ denote the data-complexity of differential cryptanalysis attack, i.e., the number of encryptions used in the attack. We now prove a *lower bound* on the data-complexity of a differential cryptanalysis attack on an r-round iterated cipher.

Theorem 9 *Suppose the hypothesis of stochastic equivalence is true, then, in an attack on an r-round iterated cipher by differential cryptanalysis,*

$$C_d(r) \geq 2/(p_{max}^{(r-1)} - \frac{1}{2^m - 1}) \tag{4.3}$$

where

$$p_{max}^{(r-1)} = \max_{\alpha} \max_{\beta} P(\Delta Y(r-1) = \beta | \Delta X = \alpha),$$

and where m is the block length of the plaintext. In particular, if $p_{max}^{(r-1)} \approx \frac{1}{2^m - 1}$, then a differential cryptanalysis attack cannot succeed.

Proof. If differential cryptanalysis is to succeed, the anticipated value β of the difference $\Delta Y(r-1)$ must certainly be taken on at least once more on the average than a randomly chosen value β'. Thus,

$$T p_{max}^{(r-1)} \geq \frac{T}{2^m - 1} + 1$$

is a necessary condition for success in T trials, where each trial consists in choosing a pair of plaintexts with the specified difference α and encrypting these two plaintexts. □

From inequality (4.3) we see that when $p_{max}^{(r-1)} \leq 3 \times 2^{-m}$, then at least 2^{m-1} pairs of encryptions are required for the attack. That implies that the required

number of (chosen or known) plaintext/ciphertext pairs is about 2^m. But there are only 2^m distinct such pairs for a fixed key so that if for almost all ciphertexts, the corresponding plaintexts are known to the attacker, then there is no need for him to determine the key since he knows the plaintext for every ciphertext. Thus, we can say that the cipher is practically secure against a differential cryptanalysis attack if $p_{max}^{(r-1)} \leq 3 \times 2^{-m}$ even if the data-complexity (2^m) is much smaller than the (processing) complexity of an exhaustive key-search attack for an iterated cipher with all the subkeys being possible.

Differentials and characteristics

In Biham and Shamir's paper [6], differential cryptanalysis of DES was described in terms of "i-round characteristics". In our notation, an i-round characteristic as defined in [6] is an $(i+1)$-tuple $(\alpha, \beta_1, ..., \beta_i)$ considered as a possible value of $(\Delta X, \Delta Y(1), ..., \Delta Y(i))$. Thus, a one-round characteristic coincides with a one-round differential and an i-round characteristic determines a sequence of i differentials, $(\Delta X, \Delta Y(j)) = (\alpha, \beta_j)$. The probability of an i-round characteristic is defined in [6] as

$$P(\Delta Y(1) = \beta_1, \Delta Y(2) = \beta_2, .., \Delta Y(i) = \beta_i | \Delta X = \alpha)$$

where the plaintext X and the subkeys $Z^{(1)}, ..., Z^{(i)}$ are independent and uniformly random. We use the notion of differentials rather than characteristics because, in the differential cryptanalysis of an r-round cipher, only the knowledge of $\Delta Y(r-1)$ is required for determining the subkey $Z^{(r)}$, no matter what the intermediate differences $\Delta Y(j), 1 \leq j < r-1$, may be. Thus, by using differential probabilities rather than characteristic probabilities, we consider in fact the true probability that differential cryptanalysis will succeed, not just a lower bound on this probability. This is why we were able to derive a *lower bound* on the complexity of a differential cryptanalysis attack from the probability of differentials. On the other hand, the probability of characteristics provides an *upper bound* on the data-complexity of the attack. Especially, for a Markov cipher whose differential probability can be well approximated by the characteristics probability (which is the case for DES with a small number of rounds), use of characteristics is more practical because that the probability of an i-round characteristic can be easily computed from the probabilities of one-round characteristics.

For a Markov cipher with independent and uniformly random round subkeys, the probability of an r-round characteristic is given by the Chapman-Kolmogorov

equation as

$$P(\Delta Y(1) = \beta_1, ..., \Delta Y(r) = \beta_r | \Delta X = \beta_0) = \prod_{i=1}^{r} P(\Delta Y(1) = \beta_i | \Delta X = \beta_{i-1}).$$

It follows from equation (4.1) that the probability of an r-round differential (β_0, β_r) is

$$P(\Delta Y(r) = \beta_r | \Delta X = \beta_0) = \sum_{\beta_1} \sum_{\beta_2} \cdots \sum_{\beta_{r-1}} \prod_{i=1}^{r} P(\Delta Y(1) = \beta_i | \Delta X = \beta_{i-1})$$

where the sums are over all possible values of differences between distinct elements, i.e., over all group elements excepting the neutral element e.

4.4 Security of Markov Ciphers

For the differential cryptanalysis of iterated ciphers, it is crucial to determine the probabilities of differentials. For a Markov cipher, such probabilities are uniquely determined by its transition matrix. In the following, we discuss the security of a Markov cipher by considering the transition matrix in terms of its irreducibility, its eigenvalues and its symmetry.

4.4.1 When is a Markov cipher secure?

Recall that a finite Markov chain with transition matrix $\Pi = [p_{ij}], 1 \leq i, j \leq M$, is said to be *irreducible* if for any (i, j), there is an r, such that the (i, j) entry in the r-th transition matrix Π^r, $p_{ij}^{(r)} > 0$. The chain is said to be *aperiodic* if

$$\gcd\left(r_i = \min_r \{p_{ii}^{(r)} > 0\};\ 1 \leq i \leq M\right) = 1.$$

The chain is said to have a *steady-state* distribution if there is a probability vector $(p_1, p_2, .., p_M)$, such that

$$\lim_{r \to \infty} p_{ij}^{(r)} = p_j, \qquad \text{for all } 1 \leq i, j \leq M. \tag{4.4}$$

The following results can be found in most of the standard books about finite Markov chains and about non-negative matrices, for example, in [3, 29, 44].

1. A finite Markov chain with transition matrix $\Pi = [p_{ij}]$ has a steady-state distribution if and only if there is an r, such that Π^r has no zero entry. In this case, the matrix Π is called *primitive*. The matrix Π is primitive if and only if there is an r_0 such that Π^{r_0} has a column that contains no zero entry.

2. A finite Markov chain with transition matrix $\Pi = [p_{ij}]$ is irreducible if and only if the chain has a unique stationary distribution.

3. A finite Markov chain has a steady-state distribution if and only if it is irreducible and aperiodic.

The security of iterated ciphers is based on the belief that a cryptographically "strong" function can be obtained by iterating a cryptographically "weak" function enough times. The following result shows that, for Markov ciphers with primitive transition matrices, iteration will give rise secure ciphers against differential cryptanalysis.

Theorem 10 *For a Markov cipher of block length m with independent and uniformly random round subkeys, if the semi-infinite Markov chain $\Delta X = \Delta Y(0)$, $\Delta Y(1)$, ... has a steady-state probability distribution, then this steady-state distribution must be the uniform distribution, i.e.,*

$$\lim_{r\to\infty} P(\Delta Y(r) = \beta | \Delta X = \alpha) = \frac{1}{2^m - 1} \tag{4.5}$$

for every non-neutral differential (α, β). If we assume additionally that the hypothesis of stochastic equivalence holds for this Markov cipher, then this cipher is secure against a differential cryptanalysis attack after sufficiently many rounds.

Proof. The theorem follows from the facts that the existence of a steady-state probability distribution implies that a homogeneous Markov chain has a unique stationary distribution, which is the steady-state distribution, and that, according to Theorem 5, the uniform distribution is a stationary distribution. □

If (4.5) holds for a Markov cipher, then the chain of differences must be irreducible and aperiodic. A sufficient condition for the chain of differences to be aperiodic is that there is an α such that the one-round differential of the form (α, α) has a non-zero probability. Based on a result in [6, p.36], we found that DES has such a value, namely (in hexadecimal representation) $\alpha = (19600000, 19600000)_x$. For the IDEA cipher, such an α also exists (see page 61). These facts imply that an essential condition for these ciphers to satisfy (4.5) is the *irreducibility* of the chain of differences. Moreover, we have the following result.

Theorem 11 *For a Markov cipher of Type III or IV considered in Section 2.4.2, the chain of differences is irreducible if and only if, for every plaintext pair (x, x^*) and every ciphertext pair (y, y^*), there is an integer r_0 and a choice of subkeys for the first r_0 rounds such that, under the first r_0 rounds of the cipher with the chosen subkeys, x is encrypted to y and x^* is encrypted to y^*.*

Proof. Recall that the round function of the ciphers in consideration has the form

$$f(X, Z) = g(X \otimes Z_A, Z_B),$$

where $\otimes$ is also the group operation for defining difference (see page 44). Let $S = X \otimes Z_A^{(1)}$ and $S^* = X^* \otimes Z_A^{(1)}$, then $\Delta S = \Delta X$.

Suppose that the chain of differences is irreducible. For the given plaintext pair (x, x^*) and ciphertext pair (y, y^*), let α be the difference of x and x^* and β be the difference of y and y^*. It follows from the irreducibility that there is an r_0 such that

$$P(\Delta Y(r_0) = \beta | \Delta S = \alpha) = P(\Delta Y(r_0) = \beta | \Delta X = \alpha) > 0,$$

which implies that there exist subkeys $z_B^{(1)}, z_A^{(2)}, z_B^{(2)}, \ldots, z_A^{(r_0)}, z_B^{(r_0)}$, and s, s^* with $\Delta s = \alpha$, such that, under these subkeys, s yields y and s^* yields y^*. Let $z_A^{(1)} = x \otimes s^{-1}$, then $s^* = x^* \otimes z_A^{(1)}$ because $\Delta s = \Delta x$. Thus, under the chosen subkeys $z_A^{(1)}, z_B^{(1)}, z_A^{(2)}, z_B^{(2)}, \ldots, z_A^{(r_0)}, z_B^{(r_0)}$, the plaintext pair (x, x^*) will yield the ciphertext pair (y, y^*).

The converse is obvious and holds in fact for all Markov ciphers. □

From this theorem, we see that, for a Markov cipher of Type III or IV, if the transition matrix is primitive, then there is an r_0 such that, for every plaintext pair (x, x^*) and every ciphertext pair (y, y^*) and for every $r \geq r_0$, there is a choice of subkeys such that, the r-round cipher function with the chosen subkeys can map the pair (x, x^*) to the pair (y, y^*). If the Markov chain is irreducible but not primitive, i.e., if the chain is periodic, then for every r, there exist a plaintext pair (x, x^*) and a ciphertext pair (y, y^*) such that, for all possible subkeys, the resulting r-round encryption function *cannot* map the pair (x, x^*) to the pair (y, y^*).

4.4.2 The number of iterations

If the transition matrix of a Markov cipher is primitive, then the cipher is secure against differential cryptanalysis attacks after sufficiently many rounds. In practice, however, only a few rounds can be used because of the requirement of ease of implementation (e.g., DES has 16 rounds and our IDEA cipher has 8 rounds). In this section, we show that the number of rounds required for a Markov cipher to be secure can be essentially determined by the the second largest eigenvalue of the transition matrix.

Let Π be the transition matrix of a Markov cipher. An *eigenvalue* of the matrix Π is a real number λ, for which there is a vector (called an *eigenvector* for λ) $\mathbf{v} = [v_1, ..., v_M]$ such that $\mathbf{v}\Pi = \lambda\mathbf{v}$. From Theorem 5, we see that 1 is an eigenvalue

of the matrix Π and that the uniform probability vector $[1/M, ..., 1/M]$, $M = 2^m - 1$, is the corresponding eigenvector. It follows from the Perron-Frobenius theory [26, 44] on non-negative matrices that if Π is primitive, then all the other eigenvalues of Π have magnitudes strictly smaller than 1 and $p_{ij}^{(r)} - 1/M$ approaches zero *exponentially* in r as r approaches infinity. For example [26, p.123], suppose that the eigenvalue λ_2 having the second largest magnitude is of multiplicity t_0 and that the eigenvalues with smaller magnitudes are of multiplicity at most t_0, then there is a constant $a > 0$, such that for all (α, β)

$$P(\Delta Y(r) = \beta | \Delta X = \alpha) - \frac{1}{2^m - 1} \leq a r^{t_0 - 1} |\lambda_2|^r. \tag{4.6}$$

From this inequality and (4.3) together with the fact that data-complexity is upper-bounded by 2^m, it follows that, if r_0 is the smallest integer such that

$$a {r_0}^{t_0 - 1} |\lambda_2|^{r_0} \leq 2^{m-1}, \tag{4.7}$$

then the Markov cipher is practically secure against a differential cryptanalysis attack after r rounds for all $r > r_0$.

4.4.3 Non-symmetry of the transition matrix

Although the irreducibility and the eigenvalues of the transition matrix are of essential importance for the security of a Markov cipher, it is difficult to determine these quantities for a cipher of large size (e.g., when m=64, the transition matrix is of size $(2^{64} - 1) \times (2^{64} - 1)$). A more practical requirement for security, which we proposed in [32] as a design principle for Markov ciphers, is the following:

The transition probability matrix of a Markov cipher should be non-symmetric.

Suppose the transition matrix Π is symmetric. Then, every pair of one-round differentials of the form (a, b) and (b, a) will have the same probability $p_{ab} = p_{ba}$. Suppose that (a, b) is the most probable one-round differential. Then $(a, b, a, b, ..., a, b, a)$ will be the the most probable $2i$-round characteristic. For small i and for the case that p_{ba} is significantly large, (a, a) will be a $2i$-round differential with high probability. This high probability can be well approximated by the probability of the corresponding $2i$-round characteristic, i.e., $p_{aa}^{(2i)} \approx p_{ba}^{2i}$. Similarly, (a, b) will be a $(2i + 1)$-round differential with high probability. This high probability can be approximated by the probability of the most probable $(2i + 1)$-round characteristic $(a, b, a, b, ..., a, b)$, i.e., $p_{ba}^{(2i+1)} \approx p_{ba}^{2i+1}$. Thus, the concatenation of the most probable one-round differential with itself $r - 1$ times produces the most probable r-round

characteristic, which will tend to provide an r-round differential with high probability. The non-symmetry of the transition matrix prevents such concatenation of highly probable one-round differentials.

Chapter 5

Differential Cryptanalysis of the IDEA Cipher

In this chapter, differential cryptanalysis of the IDEA cipher will be considered. In Section 5.1, we study the properties of the transition matrices of the mini-ciphers IDEA(m) for m=8, 16 and 32 as well as the standard IDEA(64) cipher. Under the proper choice of the group operation (5.1) for defining difference, IDEA(m) is a Markov cipher for m=8, 16, 32 and 64. The transition matrix of IDEA(m) for m=8, 16, 32 and 64 is shown to be non-symmetric. Experimental results for the IDEA(m) mini-ciphers are shown in Section 5.1.1. For m = 8 and 16, the transition matrices of IDEA(m) are further shown to be primitive. We conjecture that the transition matrices for IDEA(32) and IDEA(64) are also primitive. In Section 5.2, we consider the high-probability differentials of the IDEA cipher. Based on the properties of the three chosen group operations, the properties of the MA-structure within the cipher function, the properties of the transition matrix and the results of numerical experiments on mini IDEA ciphers and on the standard IDEA cipher, three classes of highly probable differentials of the IDEA cipher are determined. By the obtained results, the security of the IDEA cipher against differential cryptanalysis is analyzed in Section 5.3. For the plausibly most-probable differentials, which we call the "weak-key" differentials, their occurrence depend highly on the use of the "weak" keys so that such differentials are not useful for differential cryptanalysis; For the other high-probability differentials whose occurrences are (relatively) key-independent, no 3-round DC-useful differential has been found to have a probability significantly larger than 2^{-m}. We conclude that the IDEA cipher is secure against a differential cryptanalysis attack after only 4 of its 8 rounds.

5.1 Transition Matrices of IDEA(m)

In the block cipher IDEA, three group operations on $n = 16$ bit subblocks are used, namely, $\bigoplus$, bitwise XOR; $\boxplus$, addition modulo 2^n; and $\bigodot$, multiplication modulo 2^n+1 with the all-zero block representing the integer 2^n. Note that these three operations are group operations also for $n = 2$, 4 and 8. Thus, we can consider "mini" IDEA ciphers. A mini IDEA cipher has the same computational graph as the standard IDEA shown in Fig.3.1, but the subblocks are only n bits long (n=2, 4 or 8) rather than 16, and the three group operations are defined on n-bit subblocks. The resulting mini IDEA ciphers have block-length $m = 4n$. They will be denoted as IDEA(m) for $m = 8$, 16 and 32; the standard IDEA will similarly be denoted as IDEA(64).

Remark. We will not consider the case $n = 1$, i.e., we will not consider the IDEA(4), because for $n = 1$ the three operations are related as follows:

$$x \boxplus y = x \bigoplus y, \qquad x \bigodot y = x \bigoplus y \bigoplus 1, \qquad \text{for all } x, y \in \mathbb{F}_2,$$

so that the resulting cipher function of IDEA(4) is affine over GF(2). In particular, for the difference $\Delta X = X \bigoplus X^*$, every differential for IDEA(4) has probability either 1 or 0.

In the following discussion, we consider the encryptions of a pair of plaintexts by an r-round IDEA(m) cipher as shown in Fig.5.1. We assume that all the subkeys $Z^{(i)}$, $1 \le r \le r+1$, are independent and uniformly random. The round function of IDEA(m) has the form:

$$f(X, Z) = P_I(In(X \otimes Z_A, Z_B)),$$

where $\otimes$, defined below in (5.1), is a group operation on $\mathbb{F}_2^m$, where $In(\cdot,\cdot)$ is an involution cipher, i.e., for every choice of the subkey Z_B, the function $In(\cdot, Z_B)$ is an involution, and where $P_I(\cdot)$ is an involutary permutation of $\mathbb{F}_2^m$, which is also an automorphism of the group $(\mathbb{F}_2^m, \otimes)$. The computational graph of the round function is also shown in Fig.5.3 on page 65. Theorem 6 (see page 44) immediately implies the the following result.

Property 1 *For $m = 8, 16, 32$ and 64, the m-bit block cipher IDEA(m) is a Markov cipher for the difference $\Delta X = X \otimes X^{*-1}$, where the operation $\otimes$ is defined on m-bit blocks by*

$$X \otimes X^* = (X_1 \bigodot X_1^*, X_2 \boxplus X_2^*, X_3 \boxplus X_3^*, X_4 \bigodot X_4^*) \tag{5.1}$$

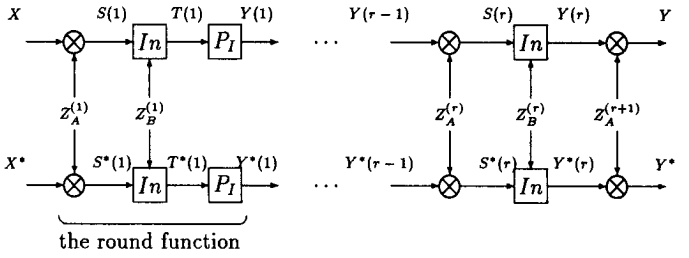


Figure 5.1: Encryptions of a pair of plaintexts by an r-round IDEA(m).

and where X^{*-1} denotes the inverse of X^* under the group operation $\otimes$.

Under the operation $\otimes$ defined in (5.1), the neutral element of the group $(\mathbb{F}_2^m, \otimes)$ is $e = (1, 0, 0, 1)$. It follows from Theorem 5 that the sequence of differences $\Delta Y(0) = \Delta X$, $\Delta Y(1), \ldots, \Delta Y(r-1)$ forms a homogeneous Markov chain when plaintexts and all subkeys are chosen independently and uniformly at random. Let Π be the transition probability matrix of this chain. We now show some properties of the transition matrix Π of IDEA(m) that in fact hold for all iterated ciphers of Types II and IV as defined in Section 2.4.2. Recall that the iterated ciphers of Type II are the iterated ciphers with round function consisting of an involution cipher and an involutary permutation. Iterated ciphers of Type IV are the iterated ciphers with round function consisting of a group cipher, an involution cipher and an involutary permutation that is also an automorphism of the group $(\mathbb{F}_2^m, \otimes)$.

Property 2 *For every iterated cipher, for every choice of the group operation for defining difference, for every α and β and for every integer $1 \leq i \leq r$,*

$$P(\Delta Y(i) = \beta | \Delta X = \alpha) = P(\Delta Y(i) = \beta^{-1} | \Delta X = \alpha^{-1}) \tag{5.2}$$

where α^{-1} denotes the inverse element of α with respect to the group operation $\otimes$ for defining difference.

Proof. For every group operation $\otimes$ defined on $\mathbb{F}_2^m$, $X \otimes X^{*-1} = \alpha$ if and only if $X^* \otimes X^{-1} = \alpha^{-1}$. □

Property 3 *For every iterated cipher of Types IV and for the difference defined by the group operation $\otimes$ used for the group cipher of the round function; and for every Markov cipher of Types II, for which the involutary permutation in the round*

function, P_I, is an automorphism of the group $(\mathbb{F}_2^m, \otimes)$ for the group operation $\otimes$ used to defined difference,

$$P(\Delta Y(i) = \beta | \Delta X = \alpha) = P(\Delta Y(i) = P_I(\alpha) | \Delta X = P_I(\beta)) \tag{5.3}$$

holds for every α and β, and for every integer $1 \leq i \leq r-1$.

Proof. To show (5.3), first consider the case $i = 1$. From Fig.5.1, which applies to any cipher of Types II and IV, we see that $\Delta X = \Delta S(1)$. Because P_I is an automorphism for the group $(\mathbb{F}_2^m, \otimes)$,

$$\Delta Y(1) = P_I(T(1)) \otimes (P_I(T(1)^*))^{-1} = P_I(T(1) \otimes (T(1)^*)^{-1}) = P_I(\Delta T(1)).$$

For any value z of subkey Z_B, the function $In(\cdot, z)$ computing $T = In(S, z)$ is an involution so that, for every key z, an input pair (S, S^*) yields the output pair (T, T^*) if and only if the input pair (T, T^*) yields the output pair (S, S^*). We have

$$P(\Delta T = \beta | \Delta S = \alpha) = P(\Delta T = \alpha | \Delta S = \beta).$$

[This implies that the transition matrix for an iterated cipher of Type I or III considered in Section 2.4.2 is always symmetric, where we recall that a cipher of Type I has a round function that is an involution cipher and that a cipher of Type III has a round function consisting of a group cipher and an involution cipher.]

Thus,

$$\begin{aligned} & P(\Delta Y(1) = P_I(\alpha) | \Delta X = P_I(\beta)) \\ = \; & P(\Delta T(1) = \alpha | \Delta S(1) = P_I(\beta)) \\ = \; & P(\Delta T(1) = P_I(\beta) | \Delta S(1) = \alpha) \\ = \; & P(\Delta Y(1) = \beta | \Delta X = \alpha), \end{aligned}$$

which shows that (5.3) hold for i=1. It now follows from (4.1) that (5.3) holds for any $1 < i \leq r-1$. □

The following lemma is useful in checking for the existence of steady-state distribution for the IDEA(m) cipher.

Lemma 4 *For any Markov cipher whose transition matrix satisfying (5.3), if there is an α in $\mathbb{F}_2^m - \{e\}$ and an integer r_0 such that, for all β in $\mathbb{F}_2^m - \{e\}$,*

$$p_{\alpha\beta}^{(r_0)} = P(\Delta Y(r_0) = \beta | \Delta X = \alpha) > 0,$$

i.e., if the r_0-th power of the transition matrix Π has a row containing no zero entry, then the Markov chain $\Delta Y(0)$, $\Delta Y(1)$,...,$\Delta Y(r-1)$ has a steady-state distribution.

Proof. We recall that a Markov chain has a steady-state distribution if and only if, for some r_0, the r_0-th transition matrix, Π^{r_0}, of the chain has a column containing no zero entry. Equation (5.3) implies that the *column* indexed by $P_I(\alpha)$ has no zero entry if and only if the *row* indexed by α has no zero entry. □

In the following discussion, we will denote the m-bit block X of IDEA(m) as $X = (X_1, X_2, X_3, X_4)$ or simply as $X = (X_1X_2X_3X_4)$, where X_i is the integer corresponding to the n-bit subblock considered as a radix-two form with the most significant bit being the leftmost bit. For example, for m=8, the 8-bit block (00011011) will be denoted as $(0, 1, 2, 3)$, or simply as (0123).

Property 4 *For IDEA(m) with m=8, 16, 32 and 64,*

$$P(\Delta Y(1) = (0110) | \Delta X = (0110)) > 0, \tag{5.4}$$

which implies that the Markov chain $\Delta Y(0)$, $\Delta Y(1)$,...,$\Delta Y(r-1)$ for the IDEA(m) cipher is aperiodic (cf. Section 4.4.1).

Proof. Consider the subkey $Z_A = e = (1, 0, 0, 1), Z_B = (0, 1)$. Under this subkey, the plaintexts $X = (0, 1, 1, 1)$ and $X^* = (1, 0, 0, 0)$ remain invariant under the round function. Thus, the one-round differential $(\Delta X = (0110), \Delta Y(1) = (0110))$ has a non-zero probability. □

In Section 4.4.3, we established a design principle for Markov ciphers, viz., that the transition matrix of a Markov cipher should be non-symmetric. The following result shows that Markov cipher IDEA(m) has a non-symmetric transition matrix.

Property 5 *For $m = 4n$ =8, 16, 32 and 64, the transition matrix of the IDEA(m) cipher is not symmetric because*

$$P(\Delta Y(1) = (1010) | \Delta X = (1100)) = 2^{-2(n-1)},$$

$$P(\Delta Y(1) = (1100) | \Delta X = (1010)) = 0.$$

These two equations will be proved below in Section 5.2.2.

5.1.1 Experimental results for mini IDEA ciphers

In order to check the existence of the steady-state distribution and to find the differentials with high probabilities for the IDEA cipher, exhaustive experiments were performed for the mini-ciphers IDEA(8) and IDEA(16). The results for the mini-ciphers suggest the behavior of the standard IDEA(64) cipher.

Existence of steady-state distribution

For the IDEA(8) cipher, the 255×255 transition matrix Π and its i-th power were computed. The results show that:

- The matrix Π^2 has 57 rows with no zero entry. For example, the row indexed with $\alpha = (3333)$ has no zero entry.
- The matrix Π^3 has no zero entry.

For IDEA(16), direct calculation has shown that the 65535×65535 matrix Π^2 has rows with no zero entry. For example, the row indexed with $\Delta X = (15, 15, 15, 15)$ has no zero entry.

Thus, by Lemma 4, we have the following result.

Property 6 *The chains of differences for IDEA(8) and for IDEA(16) have steady-state distributions.*

It now follows from Theorem 10 that IDEA(8) and IDEA(16) are secure against differential cryptanalysis after sufficiently many rounds.

For IDEA(32) and IDEA(64), such direct computations are infeasible. However, from our analysis of the cipher structure and from the results of our numerical experiments, it seems a safe conjecture that the chains of differences for IDEA(32) and for IDEA(64) also have steady-state distributions.

High-probability differentials of IDEA(8)

The i-round differentials of IDEA(8) that have high probabilities are listed in the following table. Note that, by using (5.2) and (5.3), one can also derive other differentials with high probabilities. Note also that the differentials listed in three rows of the table correspond to the three classes of high-probability differentials of IDEA(m) that will be analyzed in the following sections.

ΔX	$\Delta Y(1)$	$p^{(1)}$	$\Delta Y(2)$	$p^{(2)}$	$\Delta Y(3)$	$p^{(3)}$	$\Delta Y(4)$	$p^{(4)}$
0011	0101	2^{-2}	0000	2^{-4}	1221	$2^{-5.8}$	1202	$2^{-6.6}$
0100	1330	2^{-3}	0010	2^{-5}	1031	2^{-6}	0001	2^{-7}
1221	1202	2^{-2}	1022	2^{-3}	1221	2^{-4}	1202	$2^{-5.7}$

Table 5.1: Some differentials for IDEA(8) with high probabilities.

As discussed in Section 4.4.2, for a Markov cipher with primitive transition matrix to be practically secure in the sense that every (r-1)-round differential has

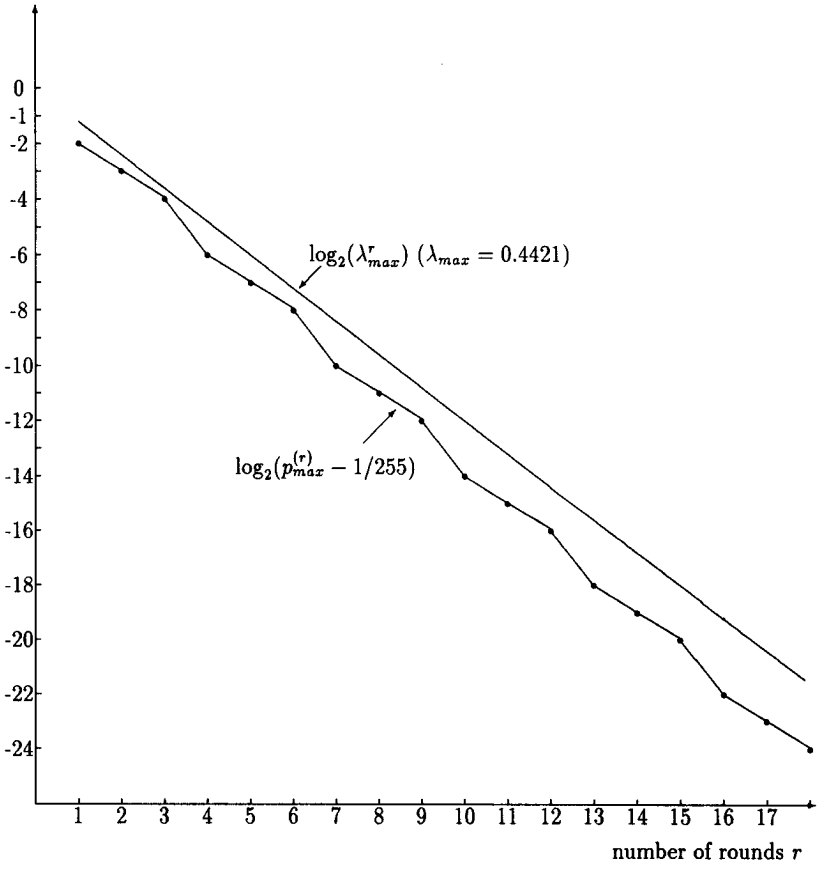


Figure 5.2: The probabilities of the most probable r-round differentials and λ^r_{max} for IDEA(8).

a probability about $2^{-(m-1)}$, the number of rounds needed is closely related to the magnitudes of the eigenvalues of the transition matrix Π with magnitudes smaller than one. For IDEA(8), all eigenvalues of Π have been computed [25] and those with large magnitudes are:
1, 0.442097, $-0.228028 + 0.365723j$, $-0.228028 - 0.365723j$, 0.402271, -0.386734, $-0.194189 + 0.330635j$, $-0.194189 - 0.330635j$, 0.368268, $-0.173838 + 0.31517j$, $-0.173838 - 0.31517j$, $\cdots$.

Let λ_{max} denote the eigenvalue of Π with maximum magnitude less than one and let $p_{max}^{(r)}$ denote the maximum probability of an r-round differential. For IDEA(8), $\lambda_{max} = 0.442097$. The $p_{max}^{(r)}$ and λ_{max}^{r} of IDEA(8) are shown in Fig.5.2. It can be clearly seen from Fig.5.2 that the maximum probability of an r-round differential decreases exponentially with the number of rounds. Note that in Fig.5.2, for $r = 6$, $p_{max}^{(r)} - 1/255 = 0.97/255 < 1/256$. Our lower bound (4.3) thus shows that the data-complexity of differential cryptanalysis attack on the 7-round IDEA(8) cipher ($> 2 \times 256$) is already larger than the total number (256) of plaintext/ciphertext pairs needed to determine the encryption function completely.

Remark. The evidence provided by the experimental results about the computing search for the most-probable differentials of IDEA(16) has shown that there are three kinds of differentials with probabilities significantly higher than the other differentials. Our numerical experiments on the IDEA(32) and on the IDEA(64) also just confirmed this hypothesis. As these three class of differentials will be discussed for all IDEA(m) cipher in detail in the following section, we will not list the experimental results. Some of the experimental results obtained for IDEA(16) can be found in [25].

5.2 High-Probability Differentials of IDEA

Numerical experiments on the IDEA mini-ciphers and theoretical analysis of the cipher structure have shown that there are three classes of differentials that could be interesting in differential cryptanalysis of the IDEA cipher. In this section, these differentials will be studied according to their special properties.

In the following discussion, we use the notation shown in Fig.5.3, where the round function of IDEA(m) is illustrated and where each variable denotes an n-bit subblock for $n = 2, 4, 8, 16$. We use the integer represented by the n-bit subblock with the most significant bit on the left, for instance, $1 = (0, ..., 0, 1)$, $2^{n-1} = (1, 0, ..., 0)$, etc. For an n-bit subblock a, let a^{-1} be the inverse of a in $(\mathbb{F}_2^n, \odot)$, let $-a$ be the inverse

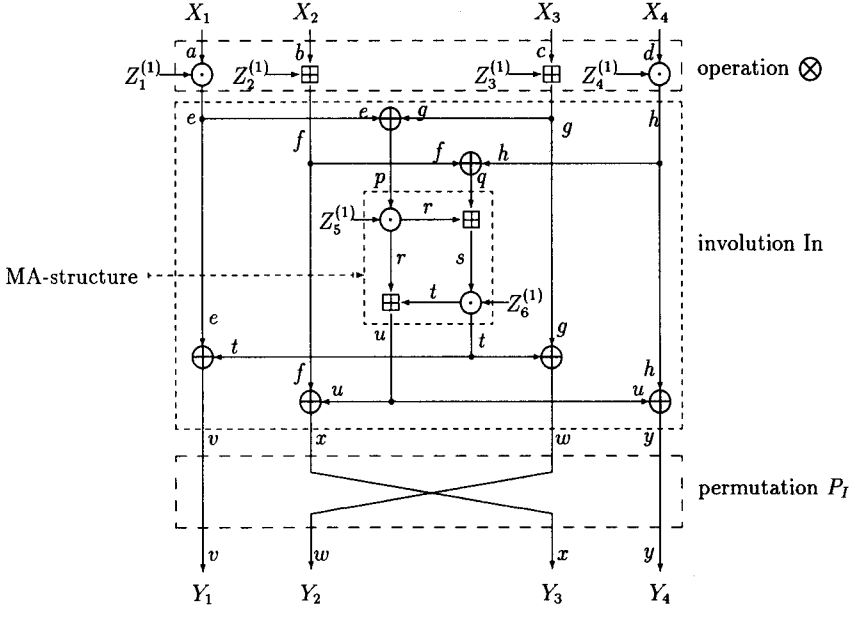


Figure 5.3: The round function of IDEA(m) and notation used for differential cryptanalysis.

of a in $(\mathbb{F}_2^n, \boxplus)$, and let $\bar{a}$ be the componentwise complement of a. For any two n-bit subblocks a and a^*, we write

$$\delta a = a \odot (a^*)^{-1}, \text{ and } \partial a = a - a^* = a \boxplus (-a^*).$$

The differences ΔX and ΔY will be expressed as $\Delta X = (\delta a, \partial b, \partial c, \delta d)$ and $\Delta Y = (\delta v, \partial w, \partial x, \delta y)$, and a one-round differential will be denoted as

$$(\Delta X = \alpha; \Delta Y = \beta) =$$
$$(\delta a = \alpha_1, \partial b = \alpha_2, \partial c = \alpha_3, \delta d = \alpha_4; \delta v = \beta_1, \partial w = \beta_2, \partial x = \beta_3, \delta y = \beta_4).$$

From Fig.5.3, we have $(\delta a, \partial b, \partial c, \delta d) = (\delta e, \partial f, \partial g, \delta h)$.

5.2.1 Transparencies of the MA-structure

The starting point of differential cryptanalysis of IDEA(m) is to determine useful relations among the three group operations. Note that the integer 0 corresponds to 2^n in $\mathbb{Z}_{2^n+1}$ and that $2^n = -1 \bmod (2^n + 1)$. Thus,

$$0 \odot a = 1 - a \qquad \text{for all } a \in \mathbb{Z}_{2^n}. \tag{5.5}$$

Because

$$a \odot a^{*-1} = 0 \iff a = 0 \odot a^* \iff a = 1 - a^* \iff a \boxplus a^* = 1,$$

it follows that, for every choice of a and a^*,

$$a \boxplus a^* = 1 \iff \delta a = a \odot (a^*)^{-1} = 0. \tag{5.6}$$

From (5.6), we obtain the following properties of the MA-structure (shown in the center of Fig.5.3) useful for differential cryptanalysis.

Theorem 12 *If the function computed by the MA-structure is written as*

$$(t, u) = MA(p, q; z_5, z_6), \tag{5.7}$$

then, for every choice of key (z_5, z_6), the inputs (p, q), (p^*, q^*) and the outputs (t, u), (t^*, u^*) of the MA-structure for the same key (z_5, z_6) satisfy the following relations:

$$\delta p = 1, \partial q = 0 \iff \delta t = 1, \partial u = 0; \tag{5.8}$$

and

$$\delta p = 0, q \boxplus q^* = 0 \iff \delta t = 0, u \boxplus u^* = 2. \tag{5.9}$$

Proof. The relation (5.8) means simply that the same inputs to the MA-structure will produce the same outputs under the same key. The relation (5.9) was first found by Murphy [32] and can be shown as follows. Because $r = p \boxdot z_5$, we have $\delta p = \delta r$. Similarly, $\delta s = \delta t$. Using (5.6), we have

$$\delta p = 0, q \boxplus q^* = 0 \iff \delta r = 0, \delta s = 0 \iff \delta t = 0, u \boxplus u^* = 2,$$

because $u \boxplus u^* = r \boxplus t \boxplus r^* \boxplus t^*$. □

By a *transparency* of the MA-structure, we mean a specific relationship between two input pairs, (p, q) and (p^*, q^*), and between the two corresponding output pairs, (t, u) and $(t,^*, u^*)$, computed with the same key (z_5, z_6), such that this relationship holds for every choice of the key (z_5, z_6). More precisely, we define a *transparency* of the MA-structure to be a 4-tuple (d_1, d_2, d_3, d_4) of invertible functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ such that, for every choice of (z_5, z_6), the equation

$$(d_3(t), d_4(u)) = MA(d_1(p), d_2(q), z_5, z_6) \tag{5.10}$$

holds for all choice of p and q when $(t, u) = MA(p, q, z_5, z_6)$.

From this definition, relation (5.8) corresponds to the transparency that is the 4-tuple of identity functions. We call this relation (5.8) the *trivial transparency* of the MA-structure. The relation (5.9), corresponding to the following 4-tuple of invertible functions:

$$d_1(p) = 0 \boxdot p,\ d_2(q) = -q,\ d_3(t) = 0 \boxdot t,\ d_4(u) = 2 - u, \tag{5.11}$$

will be called *the non-trivial transparency* of the MA-structure. The following result shows the uniqueness of this non-trivial transparency of the MA-structure.

Theorem 13 *For n=2, 4, 8 and 16, if the 4-tuple (d_1, d_2, d_3, d_4) of invertible functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ is a transparency of the MA-structure, then either $d_1 = d_2 = d_3 = d_4 =$ the identity function or else*

$$d_1(p) = 0 \boxdot p,\ d_2(q) = -q,\ d_3(t) = 0 \boxdot t,\ d_4(u) = 2 - u. \tag{5.12}$$

Proof. Suppose the 4-tuple (d_1, d_2, d_3, d_4) of invertible functions is a transparency. Note that the function of the MA-structure computes the output t as

$$t = z_6 \boxdot [q \boxplus (p \boxdot z_5)]. \tag{5.13}$$

Thus, for all p, q, z_5, z_6,

$$d_3\left(z_6 \boxdot [q \boxplus (p \boxdot z_5)]\right) = z_6 \boxdot [d_2(q) \boxplus (d_1(p) \boxdot z_5)]. \tag{5.14}$$

Letting $q = 0$ and $p = z_5 = 1$ in (5.14), we obtain

$$d_3(z_6) = z_6 \odot \delta \quad \text{for all } z_6, \tag{5.15}$$

where

$$\delta = d_2(0) \boxplus d_1(1). \tag{5.16}$$

Setting $q = 0$ and $z_6 = 1$ in (5.14) and using (5.15), we have

$$\delta \odot p \odot z_5 = d_2(0) \boxplus (d_1(p) \odot z_5) \quad \text{for all } p, z_5. \tag{5.17}$$

In (5.17), let $p = 1$ and $z_5 = 1$. Note that, for all $x, 0 \odot x = 1 - x$ so that

$$1 - \delta = d_2(0) \boxplus 1 - d_1(1).$$

Substituting (5.16) in this equation, we obtain $d_2(0) \boxplus d_2(0) = 0$. Thus, either $d_2(0) = 0$ or else $d_2(0) = 2^{n-1}$.

Suppose $d_2(0) = 2^{n-1}$. In (5.17), let $z_5 = 1$, then $d_1(p) = (\delta \odot p) - 2^{n-1}$. Substituting in (5.17) then gives

$$\delta \odot p \odot z_5 = 2^{n-1} \boxplus \left(((\delta \odot p) - 2^{n-1}) \odot z_5 \right).$$

Because $n \geq 2$, we can choose $z_5 = 2$ and choose p such that $\delta \odot p = 2^{n-1}$ in the above equation so that

$$0 = 2^{n-1} \odot 2 = 2^{n-1} \boxplus (0 \odot 2) = 2^{n-1} - 1,$$

which is valid only when $n = 1$. Thus, under the assumption $n \geq 2$, we must have $d_2(0) = 0$. In this case, letting $z_5 = 1$ in (5.17), we obtain

$$d_1(p) = \delta \odot p \qquad \text{for all } p. \tag{5.18}$$

Then, letting $z_6 = 1$ in (5.14), we have

$$\delta \odot (q \boxplus (p \odot z_5)) = d_2(q) \boxplus (\delta \odot p \odot z_5). \tag{5.19}$$

In (5.19), letting $p = z_5 = q = 1$, we obtain $\delta \odot 2 = \delta \boxplus d_2(1)$; letting $p = 0$ and $z_5 = q = 1$, we obtain $\delta = d_2(1) \boxplus 1 - \delta$. Thus, we have

$$\delta \boxplus \delta \boxplus \delta = 2 \odot \delta + 1. \tag{5.20}$$

Note that either

$$2 \odot \delta = \delta \boxplus \delta,$$

in which case we see from (5.20) that $\delta = 1$, or else

$$2 \odot \delta = \delta \boxplus \delta - 1,$$

in which case we see from (5.20) that $\delta = 0$.

For $\delta = 1$, we obtain from (5.15) and (5.18) that

$$d_1 = d_3 = \text{identity}.$$

Then (5.14) implies that d_2 must also be the identity function. Note that u is completely determined by t and r so that d_4 must also be the identity function.

For $\delta = 0$, we obtain from (5.15) and (5.18) that

$$d_1(p) = 0 \odot p, \quad d_3(t) = 0 \odot t.$$

Then, in (5.19), letting $p \odot z_5 = 0$, we obtain $d_2(q) = 0 \odot q = 1 - q$. Finally, by using (5.9), we obtain $d_4(u) = 2 - u$. □

5.2.2 Differentials based on the trivial transparency of the MA-structure

In this section we consider the one-round DC-useful differentials of the form:

$$(\alpha; \beta) = (1, o_a, 0, 0; 1, 0, o_b, 0) \text{ or } (0, 0, o_a, 1; 0, o_b, 0, 1) \tag{5.21}$$

where o_a and o_b denote odd integers between 1 and $2^n - 1$, i.e., $o_a, o_b \in \{1, 3, .., 2^n - 1\}$. The probability of differentials in this class is about $2^{-2(n-1)}$.

For this class of differentials, the input difference α was chosen to maximize the probability that

$$P((\delta p, \partial q) = (1, 0) | \Delta X = \alpha). \tag{5.22}$$

The values of the output difference β plausibly maximize the differential probability for the input difference α. Note that for the trivial transparency of the MA-structure (5.8), $(\delta p, \partial q) = (1, 0)$. Thus, we refer to this class of differentials as differentials based on the trivial transparency.

The maximizing values for α were determined by using the following property, which can be checked by direct computation.

Property 7 *For n=2, 4, 8 and 16 and for α of the form (1,o_a,0,0) or (0,0,o_a,1) where o_a is an odd integer between 1 and $2^n - 1$,*

$$P((\delta p, \partial q) = (1, 0) | \Delta X = \alpha) = \max_{\sigma} P((\delta p, \partial q) = (1, 0) | \Delta X = \sigma) = 2^{-(n-1)}.$$

To determine the maximizing values of β for such α, note that

$$P(\Delta Y = \beta | \Delta X = \alpha) \tag{5.23}$$
$$= \sum_{(\delta p, \partial q) = (i_p, i_q)} P(\Delta Y = \beta | \Delta X = \alpha, (\delta p, \partial q) = (i_p, i_q))\, P((\delta p, \partial q) = (i_p, i_q) | \Delta X = \alpha)$$
$$\geq P(\Delta Y = \beta | \Delta X = \alpha, (\delta p, \partial q) = (1,0))\, P((\delta p, \partial q) = (1,0) | \Delta X = \alpha). \tag{5.24}$$

The fact that

$$P(\delta Y_1 = 1, \partial Y_2 = 0 | \Delta X = \alpha, (\delta p, \partial q) = (1,0)) = 1 \tag{5.25}$$

suggests that the first two components of β should be (1,0). That the last two components should be (o_b,0) follows from the equation

$$\begin{aligned} & P(\Delta Y = \beta | \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \\ = \; & P(\delta Y_1 = \beta_1, \partial Y_2 = \beta_2, \partial Y_3 = \beta_3, \delta Y_4 = \beta_4 | \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \\ = \; & P(\partial Y_3 = \beta_3, \delta Y_4 = \beta_4 | \delta Y_1 = \beta_1, \partial Y_2 = \beta_2, \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \\ & \times P(\delta Y_1 = \beta_1, \partial Y_2 = \beta_2, | \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \end{aligned}$$

and from the following fact that can be checked by direct computation.

Property 8 *For n=2, 4, 8 and 16,*

$$\begin{aligned} & P(\partial Y_3 = o_b, \delta Y_4 = 0 | \delta Y_1 = 1, \partial Y_2 = 0, \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \\ = \; & \max_{\beta_3, \beta_4} P(\partial Y_3 = \beta_3, \delta Y_4 = \beta_4 | \delta Y_1 = 1, \partial Y_2 = 0, \Delta X = \alpha, (\delta p, \partial q) = (1,0)) \\ = \; & 2^{-(n-1)} \end{aligned}$$

where o_b is an odd integer between 1 and $2^n - 1$.

Thus, Properties 7 and 8 and (5.25) together suggest that the (α, β) that maximizes the right side of (5.24) should be of the form $(\alpha, \beta) = (1o_a00; 10o_b0)$. Note also that for such choices of (α, β) equality holds in (5.24). The equation (5.28) in Example 5 provides further evidence that the values of β that maximize (5.23) for α of the form $(1o_a00)$ should be $(10o_b0)$.

To compute (or to estimate) the probabilities of differentials in this class, we need the following properties of the three group operations.

$$\delta a = 0 \iff \begin{matrix} a = (A, 1\underbrace{0 \ldots 0}_{l}, \theta) \\ a* = (\overline{A}, 1\underbrace{0 \ldots 0}_{l}, \overline{\theta}) \end{matrix} \tag{5.26}$$

where A is some $[n-(l+1)]$-bit number, $l \in \{0,1,\cdots,n-1\}$ and $\theta \in \{0,1\}$.

$$\partial a = a\boxplus - a^* = 1 \iff \begin{array}{l} a = (A, 1\underbrace{0\ldots0}_{l}) \\ a* = (A, 0\underbrace{1\ldots1}_{l}) \end{array} \tag{5.27}$$

where $l \in \{0,1,\cdots,n\}$ and A is some $[n-(l+1)]$-bit number for $l < n-1$.

In the following example, we compute the probability of one specific differential in this class. By using similar, but more complicated, arguments, we can also compute the probabilities for other differentials in this class.

Example 5 *For IDEA(64), i.e., for $n = 16$,*

$$P(\Delta Y = (1010) | \Delta X = (1100)) = 2^{-30}. \tag{5.28}$$

Proof. In Fig.5.3, from $\delta e = 1$ and $\partial g = 0$, we obtain $e = e^*$ and $g = g^*$, so that $p = p^*$. From $\delta v = 1$, we have $t = t^*$ because $e = e^*$. Because $\delta r = \delta p$, we have $r = r^*$. Then from $t = t^*$, we obtain $u = u^*$. Then (5.8) implies that $\partial q = 0$. Thus, if the input difference $\alpha = (1100)$ yields the output difference $\beta = (1010)$, then $\partial q = 0$. Next, we show that

$$P(\partial q = 0 | \partial f = 1, \delta h = 0) = 2^{-15}. \tag{5.29}$$

From (5.26) and (5.27),

$$\delta h = 0 \iff h = (H, 1, 0, ..., 0, \phi) \text{ and } h^* = (\overline{H}, 1, 0, ..., 0, \overline{\phi}),$$

$$\partial f = 1 \iff f = (F, 1, 0, ..., 0) \text{ and } f^* = (F, 0, 1, ..., 1)$$

so that $h \bigoplus h^* = (1, ..., 1, 0, ..., 0, 1)$ and $f \bigoplus f^* = (0, ..., 0, 1, ..., 1)$. Note that

$$q = q^* \iff f \bigoplus h = f^* \bigoplus h^* \iff f \bigoplus f^* = h \bigoplus h^*,$$

so we must have $h \bigoplus h^* = f \bigoplus f^* = (0, ..., 0, 1)$. There are four such h's:

$$0 = (0, ..., 0), 1 = (0, ..., 0, 1), 2^{15} = (1, 0, ..., 0), 2^{15} + 1 = (1, 0, ..., 0, 1)$$

and 2^{15} such f's whose least significant bit is 1. That is, out of 2^{32} possible values of (f, h), there are exactly 2^{17} choices that will yield $\partial q = 0$, so that (5.29) holds.

Using the same argument as above, we have that for $h \in \{0, 1, 2^{15}, 2^{15}+1\}$, $f \in \{1, 3, 5, ..., 2^{16}-1\}$, $\partial f = 1$, $\delta h = 0$ and $u = u^*$,

$$(\partial x = 1, \delta y = 0) \iff u \in \{0, 2^{15}\}.$$

To summarize, we have proved that, for independent and uniformly random e, f, g, h and Z_5, Z_6, if the input difference $\alpha = (1100)$ produces the output difference $\beta = (1010)$, then

$$h \in \{0, 1, 2^{15}, 2^{15}+1\},\ f \in \{1, 3, ..., 2^{16}-1\} \text{ and } u \in \{0, 2^{15}\}. \tag{5.30}$$

Conversely, it can be easily checked that the pairs of inputs with difference $\alpha = (1100)$ and satisfying (5.30) do yield the output difference $\beta = (1010)$. Note that u takes value in $\mathbb{F}_2^{16}$ uniformly and independently of f and h, the event (5.30) happens with probability $2^{-14}2^{-1}2^{-15} = 2^{-30}$. □

For the differential of the form $(\alpha;\beta) = (0011;0101)$, one can show that its probability is essentially the same as that for (1100;1010), although slightly higher than 2^{-30}.

From the argument used above, it follows that for this class of differentials *the hypothesis of stochastic equivalence (4.2) holds exactly*, i.e., the probability of a differential conditioned on a specified value of the subkey $(Z_1, \cdots, Z_6)$ is invariant to the choice of the subkey value.

We can now prove Property 5 by showing that

$$P(\Delta Y(1) = (1100) | \Delta X = (1010)) = 0$$

for every $n \in \{2, 4, 8, 16\}$. This equation together with (5.28) imply that the transition matrix of IDEA(m) is not symmetric. From $\delta e = \delta v = 1$, we must have $t = t^*$; From $\partial f = \partial x = 0$, we must have $u = u^*$. Thus, $p = p^*$ and $q = q^*$ because, for every choice of (Z_5, Z_6), the MA structure is an invertible function from (p, q) to (t, u). However, the conditions that $\delta e = 1$ and that $\partial g = 1$ imply that $p \neq p^*$. Therefore, such a differential has zero probability.

5.2.3 Differentials based on the non-trivial transparency of the MA-structure

The differentials in this class have the form (for n=16)

$$\Delta X = (0, 1, 0, 0), \quad \Delta Y = (1, o_a, 3, 0) \text{ or } (1, o_a, 2^{16}-1, 0), \tag{5.31}$$

where o_a is in the set $\{-1, \pm 3, \pm 5, ...\}$. The probabilities of differentials in this class lie below 2^{-34}.

For this class of differentials, the input difference α was chosen to maximize the probability that

$$P(\delta p = 0, q \boxplus q^* = 0 | \Delta X = \alpha). \tag{5.32}$$

The values of the output difference β plausibly maximize the differential probability for these chosen input differences α. Note that $\delta p = 0$ and $q \boxplus q^* = 0$ correspond to the non-trivial transparency (5.9) of the MA-structure. We refer to this class of differentials as differentials based on the non-trivial transparency. This non-trivial transparency was the basis of the differential cryptanalysis (see [32]) of our earlier PES cipher.

From the fact that

$$\begin{aligned} & P((\delta p = 0, q \boxplus q^* = 0 | \Delta X = \alpha) \\ = \; & P(\delta p = 0 | \delta e = \alpha_1, \partial g = \alpha_3) P(q \boxplus q^* = 0 | \partial f = \alpha_2, \delta h = \alpha_4), \end{aligned}$$

the maximizing values for α were determined by using the following property found by direct computation.

Property 9 *For $n = 16$,*

$$P(\delta p = 0 | \delta e = \alpha_1, \partial g = \alpha_3) \approx \begin{cases} 1/3 & (\alpha_1, \alpha_3) = (0, 0) \\ 1/6 & (\alpha_1, \alpha_3) = (0, \pm 4); \end{cases}$$

and

$$P(q \boxplus q^* = 0 | \partial f = \alpha_2, \delta h = \alpha_4) \approx \begin{cases} 1/6 & (\alpha_2, \alpha_4) = (0, \pm 1) \\ 1/8 & (\alpha_2, \alpha_4) = (0, \pm 3) \\ 1/16 & (\alpha_2, \alpha_4) = (0, \pm 5) \\ 1/32 & (\alpha_2, \alpha_4) = (0, \pm 7) \\ 1/64 & (\alpha_2, \alpha_4) = (0, \pm 9), (0, \pm 11), (0, \pm 13). \end{cases}$$

[From this property and by the same argument as in [32], it can be shown that, for $n = 16$,

$$P\left(\delta v = 1, \delta w = 0, \delta x = 0, \partial y = 2^{16} - 1 | \Delta X = (0100)\right) \approx 2^{-9}. \tag{5.33}$$

Although (5.33) can be useful for cryptanalysis of the one-round IDEA, for differential cryptanalysis of IDEA, however, the proper output "difference" is to be computed as $(\delta, \partial, \partial, \delta)$, not as $(\delta, \delta, \delta, \partial)$ in (5.33). From the fact that $\delta x = 0$ will cause ∂x to take on each odd value equally likely for randomly chosen x, the differentials derived from (5.33) were found to have probabilities far below 2^{-34}.]

Property 9 implies that the input difference ΔX=(0100) maximizes (5.32). The similar estimation procedure as used in Section 5.2.2 and our numerical experiments suggested that the choice of $\alpha = (0100)$ yields differentials with higher probabilities than the other choices of α determined from Property 9. By using similar means

as in the last section, we found that the values of ΔY as shown in (5.31) plausibly maximize the one-round differential probability for $\Delta X = (0100)$. In particular, from our exhaustive experiments for IDEA(8) and IDEA(16) and our random experiments for IDEA(32) and IDEA(64) and by using similar arguments as in the last section, we conclude that the (plausibly) most probable one-round differential in this class has probability

$$P(\Delta Y = (1330)|\Delta X = (0100)) \approx 2^{-(2n+2)}$$

for n=4,8 and 16. We remark that the hypothesis of stochastic equivalence (4.2) holds also for this class of differentials.

5.2.4 Weak-key differentials

This class of differentials is closely related to the number 2^{n-1} for n=2, 4, 8 and 16. The speciality of the number 2^{n-1} is the following fact:

$$\partial x = x - x^* = 2^{n-1} \iff x \bigoplus x^* = 2^{n-1} \qquad \text{for all } x \text{ and } x^*. \tag{5.34}$$

To show this, let $x^* = (x_1, x_2, ..., x_n)$, then

$$\begin{aligned} x - x^* = 2^{n-1} &\iff x = x^* \boxplus (1, 0, ..., 0) = (\overline{x_1}, x_2, ..., x_n) \\ &\iff x = x^* \bigoplus (1, 0, ..., 0). \end{aligned}$$

The differentials in this class can be described by the following 3-round characteristic with probability $2^{-6(n-1)}$:

$$(1, 2^{n-1}, 2^{n-1}, 1) \mapsto (1, 2^{n-1}, 0, \eta) \mapsto (1, 0, 2^{n-1}, \eta') \mapsto (1, 2^{n-1}, 2^{n-1}, 1),$$

where

$$\eta, \eta' \in H = \{j;\ \text{there exist pairs of } (x, x^*),\ \text{s.t., } \delta x = j, \partial x = 2^{n-1}\}. \tag{5.35}$$

For each $\eta \in H$, let

$$S_\eta = \{a^*; a^* \boxplus 2^{n-1} = a^* \bigodot \eta\}, \tag{5.36}$$

then, for every $n \in \{2, 4, 8, 16\}$, it can be easily checked that H contains 2^{n-1} elements and that for each $\eta \in H$, S_η contains exactly two elements a^* and $0 \bigodot a^*$. For example, 2, 3, 2^{n-1} and $2^{n-1}+1$ are in the set H; and we have $S_2 = \{2^{n-1}, 2^{n-1}+1\}$, $S_{2^{n-1}} = \{3^{-1}, 0 \bigodot 3^{-1}\}$ and $S_{2^{n-1}+1} = \{0, 1\}$.

For $n = 2, 4, 8, 16$ and for $i = 1, 2, 3$, we have determined the i-round differentials of this class and their probabilities to be as follows. Firstly,

$$P\left(\Delta Y(1) = (1, 2^{n-1}, 0, \eta) | \Delta X = (1, 2^{n-1}, 2^{n-1}, 1)\right) = 2^{-2(n-1)}. \tag{5.37}$$

Moreover, this one-round differential will occur if and only if $Z_5^{(1)} \in \{0, 1\}$ and $Y_4^*(1) \in S_\eta$. Secondly,

$$P\left(\Delta Y(2) = (1, 0, 2^{n-1}, \eta) | \Delta X = (1, 2^{n-1}, 2^{n-1}, 1)\right) \approx 2^{-3(n-1)}. \tag{5.38}$$

Moreover, this differential will occur if $Z_5^{(1)}, Z_4^{(2)} \in \{0, 1\}$ and $Y_4^*(2) \in S_\eta$. Thirdly,

$$P\left(\Delta Y(3) = (1, 2^{n-1}, 2^{n-1}, 1) | \Delta X = (1, 2^{n-1}, 2^{n-1}, 1)\right) \approx 2^{-4(n-1)}. \tag{5.39}$$

Moreover, this differential will occur if $Z_5^{(1)}, Z_4^{(2)}, Z_4^{(3)}, Z_5^{(3)} \in \{0, 1\}$.

In fact, these are plausibly the most probable differentials of IDEA(m). However, the occurrence of a differential in this class is strongly key-dependent. For example, if the key subblock Z_5 is not 0 or 1 (which is the case for most keys), then the one-round differential in this class will not occur. In our terminology, the hypothesis of stochastic equivalence (see page 48) does not hold for these differentials. *Such differentials are not useful for a differential analysis attack although their probabilities are high.* We shall refer to the differentials in this class as *weak-key* differentials.

Remark. We say that 0 and 1 are "weak" key subblocks for the following reasons. First, in the encryption process for the key subblocks used as an operand for the operation $\odot$, the key value 1 has no effect on the another operand. Second, note that the multiplicative inverse of the n-tuple 0 $(=2^n)$ is itself. In particular, if the 128-bit secret key is the all-zero key and the key subblocks are generated by the key-schedule algorithm, then all the key subblocks will be zero for both encryption and decryption. The resulting encryption function is then an involution. Therefore, we refer to the all-zero 128-bit secret key as a *weak key* of the IDEA cipher.

To compute the probability for a differential in this class, we need the following property of the MA-structure.

Property 10 *For $n \in \{2, 4, 8, 16\}$ and for the MA-structure shown in the center of Fig.5.3, inputs satisfying $(\partial p, \partial q) = (2^{n-1}, 2^{n-1})$ will cause outputs satisfying $(\delta t, \partial u) = (1, 2^{n-1})$ if and only if $Z_5 \in \{0, 1\}$.*

Proof. Suppose that $Z_5 \in \{0,1\}$. In Fig.5.3, we have $r = p \odot Z_5$. If $Z_5 = 0$, then $r = 1 - p$ and $r^* = 1 - p^*$, so that

$$r - r^* = -(p - p^*) = -2^{n-1} = 2^{n-1}.$$

For $Z_5 = 1$, we have also $\partial r = \partial p = 2^{n-1}$. Thus, when $Z_5 \in \{0,1\}$, we have

$$s - s^* = (r \boxplus q) - (r^* - q^*) = \partial r - \partial q = 0,$$

which implies that $t = t^*$, i.e., $\delta t = 1$, and we have

$$\partial u = (t \boxplus r) - (t^* \boxplus r^*) = r - r^* = 2^{n-1}.$$

Conversely, suppose that inputs to the MA-structure satisfying $\partial p = 2^{n-1}$ and $\partial q = 2^{n-1}$ yield outputs satisfying $\delta t = 1$ and $\partial u = 2^{n-1}$. Then $\delta t = 1$ implies $s = s^*$, while $\partial u = 2^{n-1}$ and $t = t^*$ imply that $r - r^* = 2^{n-1}$. Because $r = p \odot Z_5$ and $p = p^* \boxplus 2^{n-1}$, then from $r = p \odot Z_5 = (p^* \boxplus 2^{n-1}) \odot Z_5$ and from $r = r^* \boxplus 2^{n-1} = (p^* \odot Z_5) \boxplus 2^{n-1}$, we have

$$(p^* \boxplus 2^{n-1}) \odot Z_5 = (p^* \odot Z_5) \boxplus 2^{n-1} \quad \text{for all } p. \tag{5.40}$$

We now show that this equation holds only if $Z_5 \in \{0,1\}$. In (5.40), letting $p^* = 1$, we have $(1 \boxplus 2^{n-1}) \odot Z_5 = Z_5 \boxplus 2^{n-1}$. Because $2 \odot (1 \boxplus 2^{n-1}) = 1$, we have

$$Z_5 = 2 \odot (1 \boxplus 2^{n-1}) \odot Z_5 = 2 \odot (Z_5 \boxplus 2^{n-1}). \tag{5.41}$$

For $n \in \{2,4,8,16\}$, it is easy to see that $Z_5 = 2^{n-1}$ cannot satisfy (5.41). If $Z_5 \in \{2^{n-1}+1, 2^{n-1}+2, ..., 2^n - 1, 2^n = 0\}$ and $Z_5' = Z_5 - 2^{n-1}$, then $Z_5' \in \{1, 2, ..., 2^{n-1}\}$ and (5.41) becomes $Z_5 = 2 \odot Z_5' = Z_5' \boxplus Z_5'$, which holds only for $Z_5' = 2^{n-1}$, i.e., $Z_5 = 2^n = 0$. For $Z_5 \in \{1, 2, .., 2^{n-1} - 1\}$, from the definition of $\odot$, (5.41) becomes (for integer addition '+')

$$\begin{aligned} Z_5 &= 2(Z_5 + 2^{n-1}) \bmod (2^n + 1) = (Z_5 + Z_5 + 2^n) \bmod (2^n + 1) \\ &= Z_5 + Z_5 - 1, \end{aligned}$$

which implies that $Z_5 = 1$. □

By using Property 10, we can now compute the probabilities of differentials of this class in the way shown in the following example.

Example 6 *For $n \in \{2,4,8,16\}$ and the set $S_{2^{n-1}}$ as defined in (5.36),*

$$P\left(\Delta Y(1) = (1, 2^{n-1}, 0, 2^{n-1}) | \Delta X = (1, 2^{n-1}, 2^{n-1}, 1)\right) = 2^{-2(n-1)}$$

and this differential will occur if and only if $Z_5 \in \{0,1\}$ *and* $Y_4^*(1) \in S_{2^{n-1}}$. *Similarly,*

$$P\left(\Delta Y(1) = (1, 0, 2^{n-1}, 2^{n-1}) | \Delta X = (1, 2^{n-1}, 0, 2^{n-1})\right) = 2^{-2(n-1)}$$

and this differential will occur if and only if h^* *and* $Y_4^*(1)$ *are in the set* $S_{2^{n-1}}$.

Proof. To show the first part, suppose that the input difference ΔX= $(1,2^{n-1},2^{n-1},1)$ yields the output difference $\Delta Y(1)$=$(1,2^{n-1},0,2^{n-1})$. Then, $\delta e = 1$ implies $e = e^*$ and $\partial g = 2^{n-1}$ implies $g \bigoplus g^* = 2^{n-1}$ because of (5.34). Thus,

$$p \bigoplus p^* = e \bigoplus g \bigoplus e^* \bigoplus g^* = 2^{n-1} \iff \partial p = 2^{n-1}.$$

Similarly, $\partial q = 2^{n-1}$. From $\delta v = 1$ and $\partial x = 2^{n-1}$, we have $t = e \bigoplus v = e^* \bigoplus v^* = t^*$, so that $\delta t = 1$. Similarly, we have $\partial u = 2^{n-1}$. Then it follows from Property 10 that $Z_5 \in \{0,1\}$. For $(\delta e, \partial f, \partial g, \delta h) = (1, 2^{n-1}, 2^{n-1}, 1)$ and $Z_5 \in \{0,1\}$, both $h = h^*$ and $\partial u = 2^{n-1}$ together imply $y - y^* = 2^{n-1}$. Thus, for $\delta y = 2^{n-1}$, y and y^* must be such that $y - y^* = y \bigodot y^{*-1} = 2^{n-1}$, i.e., $y^* \in S_{2^{n-1}}$. Conversely, for $Z_5 \in \{0,1\}$ and $y^* \in S_{2^{n-1}}$, it is easily checked that the input difference $\Delta X = (1, 2^{n-1}, 2^{n-1}, 1)$ yields the output difference $\Delta Y(1) = (1, 2^{n-1}, 0, 2^{n-1})$.

To show the second part, note that for the input difference $(1, 2^{n-1}, 0, 2^{n-1})$ to produce the output difference $(1, 0, 2^{n-1}, 2^{n-1})$, we must have $(\delta p = 1, \partial q = 0)$ and $(\delta t = 1, \partial u = 0)$. Then $\partial f = 2^{n-1}$ and $\partial q = 0$ imply $\partial h = 2^{n-1}$. But, $\delta h = \delta d = 2^{n-1}$, so that $h^* \in S_{2^{n-1}}$. Moreover, at the output, we have $\partial y = \partial h = 2^{n-1}$ and $\delta y = 2^{n-1}$, therefore, $y^* \in S_{2^{n-1}}$. The converse can be easily checked. □

Having computed the probabilities of one-round differentials in this class, we can use (4.1) to estimate the probabilities of 2-round and 3-round differentials in this class to obtain (5.38) and (5.39). Note that at the 4-th subblock, (i.e., X_4), $\partial d = 2^{n-1}$ yields $\partial h = 2^{n-1}$ if and only if $Z_4 \in \{0,1\}$ and that the set H defined in (5.35) contains 2^{n-1} values of η.

5.2.5 Differentials under the group operation XOR

Based on the results of last section, we can also consider differential under the group operation $\oplus$. The difference of two m-bit blocks is then $DX = X \oplus X^*$. (We also denote $Dh = h \oplus h^*$ for two n-bit subblocks h and h^*.) By using (5.34) and the fact that $Dh = 0 \iff \partial h = 0 \iff \delta h = 1$ and by using the same argument as in the last section, we can show that the (plausibly) most probable i-round differentials (for $i = 1, 2, 3$) of IDEA(m) under the difference DX are as follows:

Example 7 *For* $n \in \{2, 4, 8, 16\}$,

$$P\left(DY(1) = (0, 2^{n-1}, 0, 2^{n-1}) | DX = (0, 2^{n-1}, 2^{n-1}, 0)\right) = 2^{-(n-1)}$$

and this one-round differential occurs if and only if $Z_5^{(1)} \in \{0, 1\}$.

$$P\left(DY(2) = (0, 0, 2^{n-1}, 2^{n-1}) | DX = (0, 2^{n-1}, 2^{n-1}, 0)\right) \approx 2^{-2(n-1)}$$

and this two-round differential occurs if $Z_5^{(1)}$ *and* $Z_4^{(2)}$ *are both in the set* $\{0, 1\}$.

$$P\left(DY(3) = (0, 2^{n-1}, 2^{n-1}, 0) | DX = (0, 2^{n-1}, 2^{n-1}, 0)\right) \approx 2^{-4(n-1)}$$

and this three-round differential occurs if that $Z_5^{(1)}$, $Z_4^{(2)}$, $Z_4^{(3)}$ *and* $Z_5^{(3)}$ *all take values in the set* $\{0, 1\}$.

The above example shows that, for the difference defined by $\oplus$, the hypothesis of stochastic equivalence (see page 48) does not hold. Although such i-round differentials have higher probabilities than those under the difference defined by $\otimes$ as in (5.1) for i=1 and 2, their occurrence depend completely on the use of the "weak" keys. Therefore, these differentials cannot be used in a differential cryptanalysis attack.

5.3 Security of IDEA against Differential Cryptanalysis

From the discussion of Chapter 4, we know that necessary and sufficient condition for an r-round iterated cipher of block length m bits with a weak round function to be secure against a differential cryptanalysis attack is that none of the $(r\text{-}1)$-round differentials has a probability significantly higher than 2^{-m}. Moreover, for an attack by differential cryptanalysis to succeed, the high-probability differential used for the attack must be DC-useful (see 48).

For the markov cipher IDEA(m), the most-probable i-round differentials we have found so far are those shown in Section 5.2.4. In particular, IDEA(64) was shown in Section 5.2.4 to have a 3-round differential with probability about 2^{-60}. But these weak-key differentials, as shown in Section 5.2.4, occur with high probabilities only when certain key subblocks take on the "weak" values, i.e., 0 or 1. That is, these differentials are not DC-useful. For the same reason, the high-probability differentials under the operation $\oplus$ considered in Section 5.2.5 are also not useful for differential cryptanalysis of the IDEA cipher.

Note that the occurrence of a one-round differential of IDEA is always independent of the choice of the first four key subblocks Z_1, Z_2, Z_3 and Z_4. Those DC-useful one-round differentials must have probabilities that remain unchanged for all (or almost all) specified value of key subblocks Z_5 and Z_6. This implies that the corresponding pair of inputs and outputs of the MA-structure must have a certain relationship independent of Z_5 and Z_6. These facts suggest that the most probable DC-useful one-round differentials must be those that are based on the transparencies of the MA-structure. Such differentials and their probabilities have been considered in Section 5.2.2 and 5.2.3. For these differentials, no 3-round differential from these classes has been found to have a probability significantly higher than 2^{-m}, nor is any likely to exist.

Based on the above arguments, we conclude that the standard IDEA(64) cipher will be secure against a differential cryptanalysis attack after only 4 of its 8 rounds.

Chapter 6

Hash Functions Based on Block Ciphers

In cryptographic applications, block ciphers are used not only directly to provide secrecy but also indirectly to provide authenticity. In this chapter we consider the application of block ciphers in constructing hash functions. The main object of our considerations is the hash function based on iterating a round function. Section 6.2 examines the different attacks on such iterated hash functions, considers relations between the security of an iterated hash function and the security of its hash round function, and points out the wisdom of strengthening the hash function by constraining the last block of the message to be hashed.

In Section 6.3 and 6.4, we consider hash round functions constructed from secret-key block ciphers. In particular, we consider the problems of constructing m-bit hash round functions and $2m$-bit hash round functions from m-bit block ciphers. A principle is formalized for evaluating the strength of hash round functions, viz., that applying computationally simple (in both directions) invertible transformations to the input and output of a hash round function will yield a new hash round function with the same security. To demonstrate this principle, we present four attacks on three proposed $2m$-bit hash round functions. Finally, three new hash round functions based on an m-bit block cipher with a $2m$-bit key, such as the IDEA cipher, are proposed.

6.1 Hash Functions

A *hash function* is an easily implementable mapping from the set of all binary sequences of some minimum length or greater to the set of binary sequences of some fixed length. The main cryptographic applications of hash functions [45] are:

1. To provide data integrity. A message M to be protected (e.g., a data file) is transformed by the hash function into a relatively short sequence, the hash value H. This H is then stored securely. A possible modification of the message M can be detected by applying the same hash function to the purported message. The hash value H is also called a Modification (Manipulation) Detection Code (MDC) [43] or a Hash Code [22] for the message.

2. To produce short digital signatures [45]. For a user A to produce a digital signature for message M, he first computes the hash value H from the message M and then "signs" this hash value using his secret transformation S_A by computing $S_A(H)$. The obtained value $S_A(H)$ is user A's signature on the message M.

For cryptographic applications, a hash function should satisfy the first or the second of the following two security requirements:

1) It should be "one-way" in the sense that, given a hash value H, it should generally be infeasible to find a message M that hashes to this value.

2) It should be "collision-free", i.e., it should be infeasible to find two different messages that hash to the same value.

Note that if a hash function is collision-free then it is necessarily also one-way. The one-way property is necessary to prevent an interceptor of a message M and its hash value H from replacing the valid message M with a falsified message M' that he computed from the value H. The "collision-free" property is needed in digital signature schemes (and in other similar situations). If an attacker can find two different messages M and M' that hash to the same value, then he may ask an authentication center A to sign the message M and later attach A's signature to the other message M'.

6.2 Iterated Hash Functions and Attacks

An *iterated hash function* is a hash function Hash$(\cdot)$ determined by an easily computable function $h(\cdot,\cdot)$ from two binary sequences of respective lengths m and l to a binary sequence of length m in the manner that the message $M = (M_1, M_2, ..., M_n)$, where M_i is of length l, is hashed to the *hash value* $H = H_n$ by computing recursively

$$H_i = h(H_{i-1}, M_i) \qquad i = 1, 2, ..n, \tag{6.1}$$

where H_0 is a specified *initial value*. We will write $H = \text{Hash}(H_0, M)$ to show explicitly the dependence on H_0. The function h will be called the *hash round function*. Such a recursive construction of hash functions has been called the "meta-method" by Merkle [41], see also [15, 45]. For message data whose total length in bits are not multiples of l, one can apply deterministic "padding" [22, 41] to the message to be hashed by (6.1) to increase the total length to a multiple of l.

For iterated hash functions, we distinguish the following five attacks:

1. **Target attack:** Given H_0 and M, find M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.

2. **Free-start target attack:** Given H_0 and M, find H_0' and M' such that $(H_0', M') \neq (H_0, M)$ but $\text{Hash}(H_0', M') = \text{Hash}(H_0, M)$.

3. **Collision attack:** Given H_0, find M and M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.

4. **Semi-free-start collision attack:** Find H_0, M and M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.

5. **Free-start collision attack:** Find H_0, H_0', M and M' such that $(H_0', M') \neq (H_0, M)$ but $\text{Hash}(H_0', M') = \text{Hash}(H_0, M)$.

Remark. In applications where H_0 is specified and fixed, attacks 2, 4 and 5 are not real attacks. This is because the initial value H_0 is then an integral part of the hash function so that a hash value computed from a different initial value will not be accepted. However, if the sender is free to choose and/or to change H_0, attacks 2, 4 and 5 can be real attacks, depending on the manner in which the hash function is used. Note that the free-start and semi-free-start attacks are never harder than the attacks where H_0 is specified in advance. Therefore, when an iterated hash function is used in digital signature schemes in which the user may choose H_0, this initial value H_0 should also be signed.

6.2.1 Security of an iterated hash function and strength of the hash round function

For an m-bit hash function, brute-force target attacks in which one randomly chooses an M' until one hits the target $H = \text{Hash}(H_0, M)$ require about 2^m computations of hash values. It follows from the usual "birthday arguments" that brute-force collision attacks require about $2^{m/2}$ computations of hash values. In particular, for

hash round functions with $l \geq m$ so that all 2^m hash values can be reached with one-block messages, brute-force target attacks require about 2^m computations of the round function h and brute-force collision attacks require about $2^{m/2}$ computations of the round function h. We will say that the computational security of the hash function is *ideal* when there is no attack substantially better than brute force.

In the following discussion, we consider some relations between the security of an iterated hash function and the strength of its hash round function. By an *attack on the hash round function* we mean an attack in which all the involved messages contain only *one* block. For example, a target attack on the round function h reads: given H_0 and M_1, find M_1' such that $M_1' \neq M_1$ but $h(H_0, M_1') = h(H_0, M_1)$. Once a target attack on the round function yields M_1', then by "attaching" the message blocks $M_2, ..., M_n$ of the given message to M_1', one obtains success in a target attack on the iterated hash function. Similar arguments hold also for other types of attacks.

Theorem 14 *For an iterated hash function, any attack on its round function implies an attack of the same type on the iterated hash function with the same computational complexity.* □

It should be noted that the converse statement of Theorem 14 is not true in general. There may be attacks on the iterated hash function that are easier than attacks on the round function alone, as the following three examples show.

Example 8 (Long message attack.) *For an m-bit iterated hash function, given an n-block message $M = (M_1, M_2, ..., M_n)$, there is a target attack which takes about*

$$C = \begin{cases} \frac{2^m}{n} + n & \text{for } n \leq 2^{m/2} \\ 2 \times 2^{m/2} & \text{for } n > 2^{m/2} \end{cases}$$

computations of the round function. [The above result for $n \leq 2^{m/2}$ is essentially due to Winternitz [60].]

Proof. First we consider the case $n \leq 2^{m/2}$. For the given M, compute $H_i = h(H_{i-1}, M_i)$ for $i = 1, .., n$ and store these values. Then compute $H^* = h(H_0, M_1^*)$ with randomly chosen M_1^*. After computing $\frac{2^m}{n}$ values for H^*, the probability that $H^* = H_i$ for some $i, 1 \leq i \leq n$, is

$$1 - [(1 - 2^{-m})^n]^{\frac{2^m}{n}} = 1 - (1 - 2^{-m})^{2^m} \approx 1 - e^{-1} \approx 0.63,$$

which shows that usually fewer than $2^m/n$ computations of round function will be needed. The message $M' = (M_1^*, M_{i+1}, \ldots, M_n)$ hashes to the same value H as

the message M, and total number of computations of the round function is about $\frac{2^m}{n} + n$. The probability that $M' = M$ is negligible.

For $n > 2^{m/2}$, compute and store only $H_1, H_2, \ldots, H_{2^{m/2}}$. Then $2^{m/2}$ random choices of M_1^* will yield a "match" of some H^* with some $H_i, 1 \leq i \leq 2^{m/2}$, with probability about 0.63. □

For an iterated hash function, one can always do the following "trivial" free-start attacks.

Example 9 (Trivial free-start attacks.) *Consider a message $M = (M_1, M_2)$ that hashes to H with initial value H_0. Then, for the initial value H_1, the "truncated" message $M' = M_2$ hashes also to the value H. That is, a free-start target attack can always be done if the message contain more than one block. Similarly, one can do the trivial free-start collision attack.*

The following attack using a "fixed-point" of the hash round function is proposed in [46].

Example 10 (A trivial semi-free-start collision attack based on a "fixed point".) *If the hash round function h has a recognizable "fixed point", i.e., if one can somehow find a (H, M) such that $H = h(H, M)$, then there is a trivial semi-free-start collision attack since, starting with initial value $H_0 = H$, the "different" messages $M = M$ and $M' = (M, M)$ both hash to the same value H.*

Note that, in the trivial free-start and semi-free-start attacks and in the "long-message" attack described in the above three examples, one can break the iterated hash function without breaking its round function. Such attacks are based on the fact that, for an iterated hash function of the form (6.1), the attacker can take advantage of the fact that a falsified message can have a *different length* from the given genuine message. This problem can be overcome by the following strengthening of iterated hash functions, which was proposed independently by Merkle[41] and by Damgaard[15]:

Merkle-Damgaard Strengthening (MD-strengthening) *In the iterated hash function, specify that the last block M_n of the "message" $M = (M_1, M_2, ..., M_n)$ to be hashed must represent the length of the "true message" in bits, i.e., the length of the unpadded portion of the first $n - 1$ blocks.*

Using arguments similar to those in [15, 41, 47], one can show that:

Theorem 15 *Against a free-start (target or collision) attack, an iterated hash function with MD-strengthening, Hash_{MD}, has roughly the same computational security as its hash round function.*

In the previous discussions we have considered the security of an iterated hash function and the security of its round function against an attack of the *same* type. Now we consider how to use a "non-real" free-start target attack to do a "real" target attack. The following result shows that, for an iterated hash function, when a "random inverse" of the hash round function can be found with less than the ideal maximum of about 2^m computations, then there always exists a target attack on the hash function that is better than the brute-force target attack.

Theorem 16 (A meet-in-the-middle target attack by 'working backwards'.) *Let Hash_{MD} be an m-bit iterated hash function with MD-strengthening and with round function h. If, for most H in the range of h, it takes about 2^s computations of h to find a new solution (H', M) of $H = h(H', M)$ for which H' appears randomly chosen and if unconstrained portion of message contains at least two blocks, i.e., $n - 1 \geq 2$, then there exists a target attack on Hash_{MD} that takes about $2 \times 2^{\frac{m+s}{2}}$ computations of h.*

Proof. For given M and H_0, let the results of the first two iterations be

$$H_1 = h(H_0, M_1), \qquad H_2 = h(H_1, M_2).$$

We show how to find two message blocks (M_1', M_2') that hash to H_2 by a "meet-in-the-middle" attack. Then replacing the first two blocks (M_1, M_2) in the given message M by (M_1', M_2'), we obtain a message M' of the same length as, but different from, M that hashes to the same H.

First, compute $G_1 = h(H_0, M_1')$ for $2^{\frac{m+s}{2}}$ randomly chosen M_1''s; then find $2^{\frac{m-s}{2}}$ pairs (G_1'', M_2') such that $H_2 = h(G_1'', M_2')$ and G_1'' appears randomly chosen. The attack succeeds if some G_1 and some G_1'' take on the same value. Thus, the attack succeeds with probability

$$1 - [(1 - 2^{-m})^{2^{\frac{m+s}{2}}}]^{2^{\frac{m-s}{2}}} = 1 - (1 - 2^{-m})^{2^m} \approx 1 - e^{-1} \approx 0.63,$$

as follows from the facts that the probability of choosing M_1' so that G_1 will not equal G_1'' is $1 - 2^{-m}$, that there are $2^{\frac{m+s}{2}}$ independent chances to choose M_1' so that G_1 will "miss" a particular G_1'', and there are $2^{\frac{m-s}{2}}$ independently chosen values of G_1'' to miss. Both the "forwards" computation for computing values of G_1 and the

"backwards" computation for computing values of G'_1 take $2^{\frac{m+s}{2}}$ computations of the round function h. □

The method used in the above proof for attacking an iterated hash function by "working backward" [1, 59] has been used to attack several proposed iterated hash functions [45, 59]. The above result shows that if the hash round function does not have ideal computational security against a free-start target attack, then the iterated hash function cannot achieve ideal computational security against a target attack. Theorem 15, together with the argument used to prove Theorem 16, implies:

Theorem 17 *Suppose that the unconstrained portion of messages must contain at least two blocks, i.e., $n - 1 \geq 2$. Then an iterated hash function with MD-strengthening, $\text{Hash}_{\text{MD}}(\cdot)$, has ideal computational security against a target attack if and only if its hash round function $h(\cdot, \cdot)$ has ideal computational security against a free-start target attack.*

Proof. Suppose the round function h has ideal computational security against a free-start target attack. Then Theorem 15 shows that $\text{Hash}_{\text{MD}}(\cdot)$ has the same ideal security against a free-start target attack. But a target attack without free start is no easier than a free-start target attack so that $\text{Hash}_{\text{MD}}(\cdot)$ also has ideal computational security against a target attack.

Conversely, if for an m-bit hash round function h, a free-start target attack takes less than 2^m computations, then Theorem 16 implies a target attack on Hash_{MD} with less than 2^m computations. □

From the above discussions, we see that by using MD-strengthening one can obtain secure iterated hash functions from secure round functions. In particular, the trivial free-start and semi-free-start attacks and the long-message target attack shown in the above three examples *cannot* be used to attack the iterated hash function with MD-strengthening. Such considerations suggest an implementation principle for iterated hash functions, viz., that *iterated hash functions should be used only with MD-strengthening.* In the following discussion, whenever the security of an iterated hash function is considered, we always mean the security of the hash function with MD-strengthening.

Because of Theorem 17 and Theorem 15 and because one generally desires that the hash function be strong enough to provide protection against free-start attacks, the problem of constructing secure hash functions reduces to the problem of constructing hash round functions that are secure against free-start attacks, which will be considered in the next sections.

6.3 Hash Round Functions based on Block Ciphers

In the following discussion, we consider schemes for constructing hash round functions from a block cipher. In what follows, we write $Y = E_Z(X)$ for an m-bit block cipher E with k-bit key to mean that the m-bit ciphertext Y is computed from the m-bit plaintext X and k-bit key Z. Based on the discussion in the last section, we consider only attacks on the hash round function itself or on the iterated hash function with MD-strengthening.

6.3.1 Constructions of m-bit hash round functions

Davies-Meyer (DM) scheme: The DM-scheme was proposed independently by Davies and by Meyer, cf. [16, 39, 59]. This scheme can be used with any block cipher. The message block M_i that is hashed in each step of this scheme has length l to the key length k of the block cipher. The hash round function is given by

$$h(H_{i-1}, M_i) = E_{M_i}(H_{i-1}) \bigoplus H_{i-1} \tag{6.2}$$

and is illustrated in Fig.6.1 where here and hereafter $\oplus$ denotes bit-by-bit modulo-two addition.

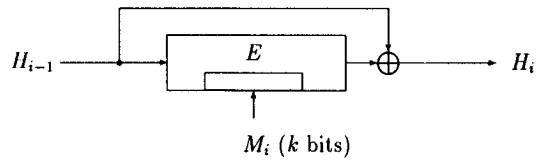


Figure 6.1: The hash round function of the DM-scheme. The small box indicates the key input to the block cipher.

The DM-scheme with MD-strengthening is generally considered to be secure in the sense that, if the block cipher has no known weakness, then no attack better than the brute-force attacks is known, i.e., the free-start target attack on h takes about 2^m computations and the free-start collision attack on h takes about $2^{m/2}$ computations. In particular, with MD-strengthening, none of the attacks mentioned in the three examples of last section can be effectively used against an iterated hash function based on the DM-scheme. The DM-scheme is currently under consideration as an ISO standard [22].

A proposed m-bit hash round function using a block cipher with m-bit block and $2m$-bit key: This method is based on a block cipher with block-length m and key-length $2m$. For example, one could use the IDEA cipher discussed in the previous chapters. For such a cipher with $k = 2m$, we will write $Y = E_{Z_a,Z_b}(X)$ to mean that the m-bit ciphertext is computed from the m-bit plaintext X and two m-bit keys Z_a and Z_b. The proposed hash round function is given by

$$h(H_{i-1}, M_i) = E_{H_{i-1},M_i}(H_{i-1})$$

and is illustrated in Fig. 6.2. We have been unable to find an attack on this hash

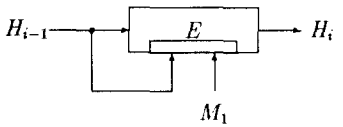


Figure 6.2: A proposed m-bit hash function based on an m-bit block cipher with a 2m-bit key.

function better than brute force attack when the underlying block cipher has no known weakness.

6.4 Construction of $2m$-bit Hash Round Function

When the block length m of a block cipher is 64 (which is the case for many practical block ciphers), one can obtain a 64-bit iterated hash function by using the DM-scheme. The "brute-force" collision attack on any 64-bit hash function has a complexity about 2^{32}, which is certainly too small in many applications. Thus, several efforts [11, 41, 43, 48, 49] have been made to construct a $2m$-bit hash function based on an m-bit block cipher by modifying the (apparently secure) DM-scheme. Before discussing this issue, we consider some relations between block ciphers and "one-way" functions.

6.4.1 Quasigroup ciphers and one-way permutations

A quasigroup cipher is a block cipher such that X, Y and Z all take values in $\mathbf{F}_2^m$ and, for every choice of the plaintext X and every choice of the ciphertext Y, there exists exactly one key Z such that $Y = E_Z(X)$.

In [19], Diffie and Hellman have pointed out that if a block cipher is secure against a known-plaintext attack, then, for a fixed plaintext X, the function $f_X(\cdot)$ defined by $f_X(Z) = E_Z(X)$ is a one-way function from the key Z to the ciphertext $Y = f_X(Z)$. This follows from the fact that to find an inverse for the function $f_X(\cdot)$ is, because X is known, equivalent to a known plaintext attack to find the key of the block cipher. If the block cipher is also a quasigroup cipher, then we have the following further result.

Theorem 18 *For a quasigroup cipher E secure against a known-plaintext attack:*
1. The function $f_X(\cdot)$ from $\mathbb{F}_2^m$ to $\mathbb{F}_2^m$ defined by $f_X(Z) = E_Z(X)$ is a "one-way" permutation, i.e., the function $f_X(\cdot)$ is one-way and invertible.
2. The function g from $\mathbb{F}_2^{2m}$ to $\mathbb{F}_2^{2m}$ defined by

$$g(X, Z) = (E_Z(X), E_Z^2(X)), \tag{6.3}$$

where $E_Z^2(X) = E_Z(E_Z(X))$, is also a one-way permutation.

Proof. The first part follows from the fact that Z is uniquely determined by given X and Y and that inverting f_X is equivalent to a known-plaintext attack on the quasigroup cipher.

To show the second part, note that knowing the pair $(E_Z(X), E_Z^2(X))$ is equivalent to knowing a plaintext/ciphertext pair so that the key Z is uniquely determined because the cipher is a quasi-group cipher. Decrypting $E_Z(X)$ with the key Z will give X so the function g is certainly invertible. Moreover, to invert the function g requires one to determine the key Z from $E_Z(X)$ ("plaintext") and $E_Z^2(X)$ ("ciphertext"), which is just a known plaintext attack on the underlying cipher. □

Remark. The method of using a quasigroup of 2^m elements to produce a permutation on a set of 2^{2m} elements has been discussed in [18]. The cryptographic interest here is that a secure m-bit quasigroup cipher can be used to obtain a "one-way" $2m$-bit permutation. Note that inverting a $2m$-bit one-way function by brute force requires 2^{2m} computations of the function, but inverting the $2m$-bit function g defined above is equivalent to attacking an m-bit block cipher with m-bit key which needs at most 2^m encryptions. Thus, the $2m$-bit function g constructed as in (6.3) from an m-bit quasigroup cipher can be only "half one-way". However, "cascading" two m-bit block ciphers to obtain a $2m$-bit one-way function could still yield a secure $2m$-bit hash function from an m-bit block cipher as will be considered in the following sections.

6.4.2 A principle for evaluating hash round functions and four attacks on three $2m$-bit hash round functions

In this section, we establish a principle for evaluating the security of a hash round function, viz. that *applying any simple (in both directions) invertible transformations to the input and to the output of the hash round function yields a new hash round function with the same security as the original one.* [A similar principle has been used by Meier and Staffelbach in [40] to classify nonlinearity criteria for cryptographic functions]. For example, for a block cipher with block length equal to key length, it follows from this principle that the hash round function (6.2) of the DM-scheme has the same security as the following hash round function proposed in [39]

$$h(H_{i-1}, M_i) = E_{H_{i-1}}(M_i) \bigoplus M_i$$

in which the hash round function differs only by a "swapping" of the blocks H_{i-1} and M_i from that in (6.2).

To demonstrate this principle, we present four "meet-in-the-middle" attacks on three $2m$-bit hash round functions based on an m-bit block cipher with an m-bit key. The basic purpose of these three schemes is to construct a $2m$-bit hash function based on an m-bit block cipher by modifying the (apparently secure) DM-scheme (6.2). We now show that these $2m$-bit hash round functions are in fact weaker than the m-bit hash round function of the DM-scheme. More precisely, for each scheme, we present a free-start target attack that takes only about $2^{m/2}$ (instead of the ideal maximum 2^{2m}) computations of the round function. [Note that the free-start target attack on the m-bit hash round function in the DM-scheme has complexity 2^m.]

The Preneel-Bosselaers-Govaerts-Vandewalle (PBGV) scheme.
The PBGV scheme was proposed in [48]. In this scheme, which uses an m-bit block cipher with an m-bit key, a $2m$-bit hash value $H = (H_n, G_n)$ is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value (H_0, G_0). In each round, two new m-bit values H_i and G_i are computed from the two previous m-bit values H_{i-1} and G_{i-1} and from the two m-bit message blocks L_i and N_i as follows:

$$\begin{aligned} H_i &= E_{L_i \oplus N_i}(H_{i-1} \oplus G_{i-1}) \oplus L_i \oplus H_{i-1} \oplus G_{i-1} \\ G_i &= E_{L_i \oplus H_{i-1}}(N_i \oplus G_{i-1}) \oplus N_i \oplus H_{i-1} \oplus G_{i-1} \end{aligned} \tag{6.4}$$

for $i = 1, 2, \ldots, n$. The round function for the PBGV-scheme produces the output pair (h, g) from the inputs (h_0, g_0, l, n) in the manner

$$\begin{aligned} h &= E_{l \oplus n}(h_0 \oplus g_0) \oplus l \oplus h_0 \oplus g_0 \\ g &= E_{l \oplus h_0}(n \oplus g_0) \oplus n \oplus h_0 \oplus g_0. \end{aligned} \tag{6.5}$$

By applying the transformation

$$(h, g) \longrightarrow (h, f) = (h, h \oplus g) \tag{6.6}$$

on the output and the transformation

$$(h_0, g_0, l, n) \longrightarrow (h'_0, g'_0, l', n') = (h_0 \oplus g_0, g_0 \oplus n, l \oplus n, n), \tag{6.7}$$

on the input, we obtain the round function illustrated in Fig.6.3 that computes (h, f) from the input (h'_0, g'_0, l', n') in the manner

$$\begin{aligned} h &= E_{l'}(h'_0) \oplus l' \oplus n' \oplus h'_0 \\ f &= E_{l' \oplus h'_0 \oplus g'_0}(g'_0) \oplus E_{l'}(h'_0) \oplus l'. \end{aligned} \tag{6.8}$$

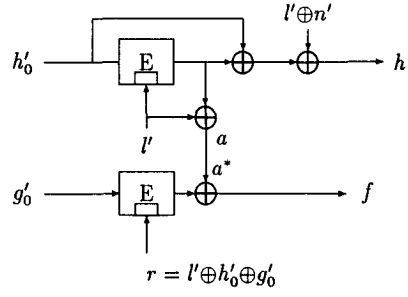


Figure 6.3: The transformed function used to attack the PBGV round function.

Because the transformations (6.6) and (6.7) are both easy to compute and easy to invert, it follows that an attack on (6.8) has the same complexity as an attack on (6.5).

A free-start target attack on the PBGV round function with complexity about $2^{m/2}$: In this attack, we show how to find a "random inverse" of (6.8), i.e., we show how, for given (h, f), to find (h'_0, g'_0, l', n') satisfying (6.5) for which (h'_0, g'_0) appears randomly chosen.

1. Choose an arbitrary constant c_0.

2. For the given h, compute $a = E_{l'}(h'_0) \oplus l'$ for $2^{m/2}$ randomly chosen values of (h'_0, l') such that $h'_0 \oplus l' = c_0$.

3. For the given f, compute $a^* = E_r(g'_0) \oplus f$ for $2^{m/2}$ randomly chosen values of (g'_0, r) such that $g'_0 \oplus r = c_0$.

The probability that some a and some a^* take on the same value is about 0.63. For such $(g'_0, r, a = a^*, h'_0, l')$, we obtain a solution (h'_0, g'_0, l', n') for (6.8) by computing $n' = a \oplus l' \oplus h'_0 \oplus l' \oplus h$. □

A target attack on the PBGV round function with complexity about 2^m: In this attack, we find, for the given (h_0, g_0) and (h, g), a message block (l, n) satisfying (6.5). We will use the notation of Fig.6.3.

From (6.6) and (6.7), we see that (h, f) and h'_0 are determined by the given (h_0, g_0) and (h, g). We randomly choose l', then compute

$$a = E_{l'}(h'_0) \oplus l',$$

$$n' = a \oplus h'_0 \oplus h,$$

$$r = l' \oplus h'_0 \oplus g'_0 = l' \oplus h'_0 \oplus g_0 \oplus n'$$

and

$$g'_0 = D_r(a \oplus f),$$

where $D_z(y)$ denotes the result of deciphering y with key z.

After 2^m such computations, $g'_0 \oplus n'$ will take on the given value g_0 with probability 0.63. Then using (6.6) and (6.7), we obtain a solution (l, n) for (6.5). □

The first Quisquater-Girault (QG-I) scheme.

The QG-I scheme was proposed in the Abstracts from Eurocrypt'89 [49]. It also appeared in a draft ISO standard [21], see also [45]. However, this scheme was dropped from the recent version of the draft ISO standard CD10118 [22]. [In unpublished work, Coppersmith pointed out some weakness of this scheme [50]. In the subsequent Proceedings paper [50], a "weaker" round function was used, but with additional functional strengthening.] Similarly to the PBGV-scheme discussed above, the QG-I scheme is based on an m-bit block cipher with an m-bit key. A $2m$-bit hash value (H_n, G_n) is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value (H_0, G_0). In each round, two new m-bit values H_i and G_i are computed from the two previous m-bit values H_{i-1} and G_{i-1} and from the two m-bit message blocks L_i and N_i as follows:

$$\begin{aligned} W_i &= E_{L_i}(G_{i-1} \oplus N_i) \oplus N_i \oplus H_{i-1} \\ H_i &= W_i \oplus G_{i-1} \\ G_i &= E_{N_i}(W_i \oplus L_i) \oplus H_{i-1} \oplus G_{i-1} \oplus L_i \end{aligned} \tag{6.9}$$

for $i = 1, 2, \ldots, n$. The round function of the QG-I scheme produces the output pair (h, g) from the input (h_0, g_0, l, n) in the manner

$$\begin{aligned} h &= E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ g &= E_n\left(E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus h_0 \oplus g_0 \oplus l. \end{aligned} \tag{6.10}$$

We will consider the pair $(h, f) = (h, h \oplus g)$ illustrated in Fig.6.4 and defined by

$$\begin{aligned} h &= E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ f = h \oplus g &= E_n\left(E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus E_l(g_0 \oplus n) \oplus l \oplus n. \end{aligned} \tag{6.11}$$

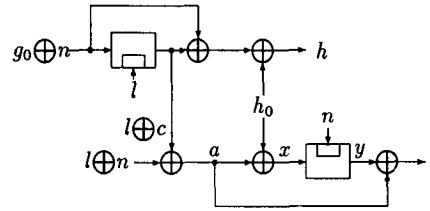


Figure 6.4: The pair (h, f) used in the attack on the QG-I scheme.

A free-start target attack on the QG-I scheme with complexity about $2^{m/2}$: In the following we show that, for any given (h, f), one can indeed find, in about $2^{m/2}$ decrypting computations for the block cipher, a solution (h_0, g_0, l, n) satisfying (6.11) by a "meet-in-the-middle" attack.

We will use the notation shown in Fig.6.4. Let c be a fixed m-tuple.

1. Randomly choose values for a and n such that $a \oplus n = c$. Then, for the given value of f, compute $h_0' = a \oplus D_n(a \oplus f)$. Repeat this process $2^{m/2}$ times to obtain $2^{m/2}$ values for (h_0', n) with randomly chosen values for h_0'.

2. Randomly choose l and compute $h_0^* = h \oplus (l \oplus c) \oplus D_l(l \oplus c)$. In $2^{m/2}$ computations, we obtain $2^{m/2}$ values for (h_0^*, l) with randomly chosen values for h_0^*.

Note that both h_0' and h_0^* are m-bit blocks so that some h_0' and some h_0^* obtained as above will take on the same value with probability about 0.63. Thus, we can find (h_0', h_0^*, l, n) such that $h_0' = h_0^*$. (Note that the constraint that $l \oplus c \oplus l \oplus n = a$ is automatically satisfied.) From the obtained (l, n), compute $g_0 = D_l(l \oplus c) \oplus n$. Then the resulting (h_0, g_0, l, n) is the desired solution. □

The LOKI Double Block Hash (DBH) function. The block cipher LOKI, proposed in [11], is a DES-like 64-bit block cipher with a 64-bit key. In [11], a 128-bit iterated Double Block Hash (DBH) function based on the cipher LOKI was also proposed, but this scheme can be used for any m-bit block cipher with an m-bit key. In LOKI DBH, a $2m$-bit hash value (H_n, G_n) is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value (H_0, G_0). In each round, two new m-bit values H_i and G_i are computed from the two previous m-bit values H_{i-1} and G_{i-1} and from the two current m-bit message blocks L_i and N_i as follows:

$$\begin{aligned} W_i &= E_{L_i \oplus G_{i-1}}(G_{i-1} \oplus N_i) \oplus N_i \oplus H_{i-1} \\ H_i &= W_i \oplus G_{i-1} \\ G_i &= E_{N_i \oplus H_{i-1}}(W_i \oplus L_i) \oplus H_{i-1} \oplus G_{i-1} \oplus L_i \end{aligned} \tag{6.12}$$

for $i = 1, 2, \ldots, n$.

The LOKI DBH round function was derived from the hash round function of the QG-I scheme (6.9) by the bitwise addition modulo 2 of the previous hash value blocks (H_{i-1} and G_{i-1}) to the current message blocks (L_i and N_i) to obtain the key inputs for the two LOKI encryptions. This was done to avoid some attacks derived from the 'weak key' of the underlying cipher. By applying our security evaluation principle, we obtain the following free-start target attack on the LOKI DBH round function that has complexity only about $2^{m/2}$.

The round function for the LOKI DBH produces the output pair (h, g) from the input (h_0, g_0, l, n) in the manner

$$\begin{aligned} h &= E_{l \oplus g_0}(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ g &= E_{n \oplus h_0}\left(E_{l \oplus g_0}(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus h_0 \oplus g_0 \oplus l. \end{aligned} \tag{6.13}$$

By applying the transformation

$$(h, f) = (h, h \oplus g) \tag{6.14}$$

on the LOKI DBH output pair (h, g) and applying the transformation

$$(h_0, g_0, l', n') = (h_0, g_0, l \oplus g_0, n \oplus g_0) \tag{6.15}$$

on the LOKI DBH inputs (h_0, g_0, l, n), we obtain the function illustrated in Fig.6.5 that computes (h, f) from the inputs (h_0, g_0, l', n') in the manner

$$\begin{aligned} h &= E_{l'}(n') \oplus n' \oplus h_0 \\ f &= E_{n' \oplus h_0 \oplus g_0}(h \oplus l') \oplus h \oplus l' \oplus h_0. \end{aligned} \tag{6.16}$$

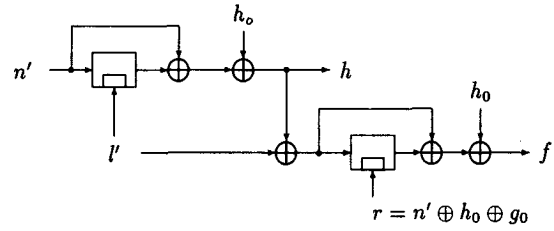


Figure 6.5: The new function used to attack the LOKI DBH round function.

A free-start target attack on the LOKI DBH with complexity about $2^{m/2}$: In the following, we show that, for any given (h, f), one can indeed find, in about $2 \times 2^{m/2}$ encrypting computations for the block cipher, a solution for (h_0, g_0, l, n) satisfying (6.11) by a "meet-in-the-middle" attack.

Because the transformations (6.14) and (6.15) are both easy to compute and easy to invert, it follows that finding a solution (h_0, g_0, l, n) of (6.13) for a given (h, g) is computationally the same as finding a solution (h_0, g_0, l', n') of (6.16) for a given (h, f). This can be done in about $2 \times 2^{m/2}$ encryptions as we now show.

1. Choose an arbitrary value for l'.

2. For the given h and the chosen l', compute $h_0 = h \oplus n' \oplus E_{l'}(n')$ for $2^{m/2}$ randomly chosen values of n'.

3. For the given h, f and the chosen l', compute $h_0^* = E_r(h \oplus l') \oplus h \oplus l' \oplus f$ for $2^{m/2}$ randomly chosen values of r $(= n \oplus h_0 \oplus g_0)$.

The probability that some h_0 and some h_0^* take on the same value is about 0.63. For $h_0 = h_0^*$, by computing $g_0 = r \oplus n' \oplus h_0$, we obtain a solution (h_0, g_0, l', n') for (6.16). □

Remark. We have shown in this section three free-start target attacks on three hash round functions. The "real" target attacks (with specified initial value) will usually be more difficult. For example, when m is 64 bits, a target attack on the 128-bit hash function LOKI DBH obtained by combining the above attack with the attack used in the proof of Theorem 3 will take about $2^{\frac{128-32}{2}} = 2^{80}$ computations. The same conclusion holds also for the QG-I scheme hash function.

6.4.3 Complexity of known attacks on some $2m$-bit hash functions

We consider here some known 128-bit iterated hash functions based on two uses of an $m = 64$-bit block cipher with key-length $k = 64$ or $k = 56$ in each round. All these schemes are obtained by modifying the 64-bit DM-scheme hash round function. The complexities of known attacks on these hash functions are listed in Table 6.1. We assume that all the iterated hash functions are used with MD-strengthening and that the underlying block cipher has no known weakness (such as weak keys).

$h(\cdot,\cdot)$	PBGV	GQ-I	LOKI-DBH	Merkle$_{\rightsquigarrow 12}$	M-S$_{\rightsquigarrow 13}$	ideal
$(m,k)_{\rightsquigarrow 1}$	(64,64)	(64,64)	(64,64)	(64,56)	(64,56)	(64,k)
target	$2^{64}_{\rightsquigarrow 2}$	$2^{80}_{\rightsquigarrow 5}$	$2^{80}_{\rightsquigarrow 9}$	2^{112}	$2^{81}_{\rightsquigarrow 14}$	2^{128}
f-s target	$2^{32}_{\rightsquigarrow 3}$	$2^{32}_{\rightsquigarrow 6}$	$2^{32}_{\rightsquigarrow 10}$	2^{112}	$2^{54}_{\rightsquigarrow 15}$	2^{128}
collision	2^{64}	2^{64}	2^{64}	2^{56}	2^{54}	2^{64}
semi-f-s col.	2^{64}	$2^{32}_{\rightsquigarrow 7}$	2^{64}	2^{56}	2^{54}	2^{64}
f-s coll.	$2^{32}_{\rightsquigarrow 4}$	$o(1)_{\rightsquigarrow 8}$	$2^{32}_{\rightsquigarrow 11}$	2^{56}	$2^{27}_{\rightsquigarrow 16}$	2^{64}
leng(M_i)	128	128	128	7	64	$l_{\rightsquigarrow 17}$

$\rightsquigarrow 1$: m: block-length, k: key-length of the underlying cipher;

$\rightsquigarrow 2, 3$: see last section;

$\rightsquigarrow 4$: a free-start collision attack is no harder than a free-start target attack;

$\rightsquigarrow 5$: from the free-start target attack$_{\rightsquigarrow 6}$ and Theorem 16;

$\rightsquigarrow 6$: see last section;

$\rightsquigarrow 7, 8$: see [46];

$\rightsquigarrow 9, 10$: same as $\rightsquigarrow 5, 6$;

$\rightsquigarrow 11$: *same as* $\rightsquigarrow 4$;

$\rightsquigarrow 12$: Merkle's scheme [41]: hash-code is of length 112 bits; this scheme appears to have ideal security; however, each round can 'digest' only 7 bits of message;

$\rightsquigarrow 13$: Meyer-Schilling's scheme [43]: 128-bit hash code, but each round output has length 108 bits;

$\rightsquigarrow 14, 15$: each round output (two blocks) has length 108 bits; a free-start target attack on one (54-bit) block takes about 2^{54} computations; then use Theorem 16; see also [43];

$\rightsquigarrow 16$: collision is achieved on one (54-bit) block.

$\rightsquigarrow 17$: see next section.

Table 6.1: Complexity of known attacks on some hash round functions.

6.4.4 Proposed schemes for block ciphers with $k = 2m$

The study of previously proposed hashing schemes (see Table 6.1) suggests that it is difficult, if not impossible, to build a $2m$-bit hash round function with ideal computational security that can "digest" in each round at least m bits of message by two uses of an m-bit block cipher with an m-bit key. However, if an m-bit block cipher with a $2m$-bit key is available, then there are more possibilities to construct secure round functions. In following, we propose two $2m$-bit hash round functions that use an m-bit block cipher with a $2m$-bit key and that appear to be secure.

Tandem DM: We refer to our first proposed $2m$-bit hash function as the *Tandem* DM scheme because it is based on cascading two DM-schemes as in (6.2). The round function of the Tandem DM scheme is shown in Fig.6.6. In each iteration, two new

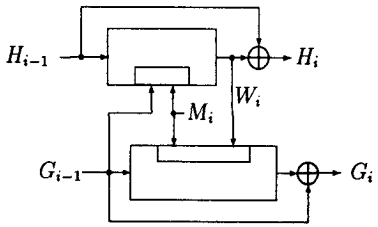


Figure 6.6: The Tandem DM $2m$-bit hash round function based on an m-bit block cipher with a $2m$-bit key.

m-bit values H_i, G_i are computed from the two previous m-bit values H_{i-1} and G_{i-1} and from an m-bit message block M_i as follows:

$$\begin{aligned} W_i &= E_{G_{i-1},M_i}(H_{i-1}) \\ H_i &= W_i \oplus H_{i-1} \\ G_i &= G_{i-1} \oplus E_{M_i,W_i}(G_{i-1}). \end{aligned}$$

Abreast DM We next propose the *Abreast* DM scheme in which two DM-schemes are used side-by-side. The hash round function is illustrated in Fig.6.7. In each round, two new m-bit values H_i, G_i are computed from the two previous m-bit values H_{i-1}, G_{i-1} and from an m-bit message block M_i as follows:

$$\begin{aligned} H_i &= H_{i-1} \oplus E_{G_{i-1},M_i}(H_{i-1}) \\ G_i &= G_{i-1} \oplus E_{M_i,H_{i-1}}(\overline{G}_{i-1}) \end{aligned}$$

where $\overline{G}$ denotes the bit-by-bit complement of G.

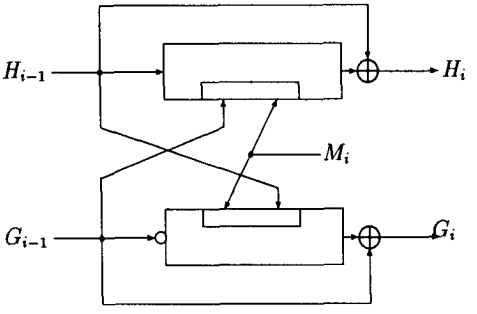


Figure 6.7: The Abreast DM $2m$-bit hash round function based on an m-bit block cipher with a $2m$-bit key. The circle indicates that the input to the lower encrypter is bitwise complemented.

Remarks: 1. The Tandem DM and the Abreast DM schemes are constructed based on the following consideration:

The round function h consists of two subfunctions h_1 and h_2:

$$(H_i, G_i) = h(H_{i-1}, G_{i-1}, M_i) = [h_1(H_{i-1}, G_{i-1}, M_i), h_2(H_{i-1}, G_{i-1}, M_i)],$$

both of which have the same inputs. Thus, to attack h (in a free-start target or free-start collision attack) implies that one must attack both h_1 and h_2 simultaneously. If the subfunctions h_1 and h_2 are so 'different' that an attack on one subfunction provides no help in attacking the other subfunction and if both h_1 and h_2 are equivalent (in the sense of security) to the apparently secure DM-scheme, then we can expect that an attack on h will have complexity equal to the product of the complexities of the attacks on h_1 and on h_2. In the proposed Tandem DM and Abreast DM schemes, the subfunctions h_1 and h_2 are chosen to be as "different" as possible.

2. The Abreast DM scheme gives a $2m$-bit hash function that is at least as strong as the m-bit DM-scheme. [The same holds also for the Meyer-Schilling's scheme [22, 43].]

3. Our investigations to this point have shown no weakness in either of these two new proposed $2m$-bit hash round functions, i.e., we have been unable to find any attacks better than brute-force attacks when the underlying cipher is assumed to have no weakness. We should point out, however, that our Tandem DM and Abreast DM schemes use two m-bit block encryptions for each block of m message bits in order to compute a final hash value of length $2m$ bits.

Chapter 7

Concluding Remarks

The challenge in the development of secret-key block ciphers lies in the fact that the design of such ciphers is presently more art than science. There is a clear need for more scientific formulation of the principles on which the security of such ciphers rests. In this work, we have considered some well-established general design principles for block ciphers and have introduced others. Shannon's principle of confusion and diffusion was adapted to meet our security requirements. Our principle of using incompatible group operations on subblocks was motivated by software implementation constraints. The principles of E/D similarity and regular structure were motivated by hardware implementation constraints. Based on these principles, we have considered the design and security of iterated block ciphers. In particular, we developed the IDEA cipher by applying these general design principles. The interaction of the three chosen "incompatible" group operations in the IDEA cipher provides the necessary "confusion", and the MA structure within the cipher causes the required "diffusion".

The security of iterated ciphers against Biham and Shamir's differential cryptanalysis has been studied. We described differential cryptanalysis in terms of an i-round "differential" instead of in terms of the i-round characteristic" used in Biham and Shamir's original paper on differential cryptanalysis. It was shown that the maximum probability of such a differential can be used to determine a lower bound on the complexity of a differential cryptanalysis attack. The concept of "Markov ciphers" was introduced because of its significance in differential cryptanalysis. It was shown that the security of a Markov cipher against differential cryptanalysis is determined by the transition probability matrix created by the round function. A new design principle for Markov ciphers was formulated, viz., that their transition matrices should be non-symmetric. The IDEA cipher was developed according to this principle from its previous version PES which had a symmetric transition ma-

trix. Our study of the differential cryptanalysis attack on the IDEA cipher suggests that the IDEA cipher is secure against a differential cryptanalysis attack after only four of its eight rounds.

Block ciphers can be used in many different modes so that the same block cipher may serve for several different cryptographic applications. In particular, block ciphers are often used in constructing iterated hash functions. The adage that "a chain is no stronger than its weakest link" is nowhere so true as for iterated hash functions. A secure iterated cipher can be obtained by iterating a suitably chosen "weak" round function; an iterated hash function, however, can never be stronger than its hash round function but can be weaker. However, with the use of Merkle-Damgaard-strengthening, a secure hash function can be obtained from a secure round function. Schemes for constructing hash round functions by using a block cipher were considered. Three hashing schemes based on the IDEA cipher were proposed. In particular, we considered the problem of constructing $2m$-bit hash round functions from m-bit block ciphers. A principle was formalized for evaluating the strength of hash round functions, viz., that applying simple (in both directions) invertible transformations to the input and output of a hash round function will yield a new hash round function with the same security. By applying this principle, we devised four attacks on three $2m$-bit hash round functions based on an m-bit block cipher.

Bibliography

[1] S. G. Akl, "On the Security of Compressed Encodings", Advances in Cryptology – CRYPTO'83, Proceedings, pp. 209-230, Plenum Press, New York, 1984.

[2] ANSI X9.17-1985, *Financial Institution Key Management (Wholesale)*, American Bankers Association, 1985.

[3] R. Ash, *Information Theory*, Interscience Publishers, 1965.

[4] H. Beker and F. Piper, *Cipher Systems*, Northwood Books, London, 1982.

[5] T. A. Berson, "Long Key Variants of DES", Advances in Cryptology – CRYPTO'82, Proceedings, pp. 311-314, Plenum Press, New York, 1982.

[6] E. Biham and A. Shamir, "Differential Cryptanalysis of DES–like Cryptosystems", Journal of Cryptology, Vol. 4, No. 1, pp. 3-72, 1991.

[7] E. Biham and A. Shamir, "Differential Cryptanalysis of FEAL and N-Hash", Advances in Cryptology – EUROCRYPT'91, Proceedings, LNCS 547, pp. 1-16, Springer-Verlag, Berlin 1991.

[8] E. Biham and A. Shamir, "Differential Cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer", Advances in Cryptology – CRYPTO'91, Proceedings, LNCS 576, pp. 156-171, Springer-Verlag, Berlin 1992.

[9] E. Biham and A. Shamir, "Differential Cryptanalysis of the full 16-round DES", Preliminary manuscript, December 19, 1991.

[10] H. Bonnenberg, A. Curiger and H. Kaeslin, "LEONARDO – Design Aspects of the VLSI Implementation of a New Secret Key Block Cipher," Technical Report 05/90, Integrated System Laboratory, ETH Zürich, April 1990.

[11] L. Brown, J. Pieprzyk and J. Seberry, "LOKI – A Cryptographic Primitive for Authentication and Secrecy Applications", Advances in Cryptology – AUSCRYPT'90, Proceedings, LNCS 453, pp. 229-236, Springer-Verlag, 1990.

[12] D. Chaum and J. H. Evertse, "Cryptanalysis of DES with a Reduced Number of Rounds", Advances in Cryptology – CRYPTO'85, Proceedings, LNCS 218, pp. 192-211, Springer-Verlag, 1986.

[13] T. W. Cusick and M. C. Wood, "The REDOC-II Cryptosystem", Advances in Cryptology – CRYPTO'90, Proceedings, LNCS 537, 1990.

[14] *Data Encryption Standard*, FIPS PUB 46, National Tech. Info. Service, Springfield, VA, 1977.

[15] I. B. Damgaard, "A Design Principle for Hash Functions", Advances in Cryptology – CRYPTO'89, LNCS 435, pp. 416-427, Springer-Verlag, 1990.

[16] R. W. Davies and W. L. Price, "Digital Signature – an Update", Proc. International Conference on Computer Communications, Sydney, Oct 1984, Elsevier, North-Holland, pp. 843-847, 1985.

[17] D. W. Davies and W. L. Price, *Security for Computer Networks*, John Wiley & Sons, 1984.

[18] J. Dénes and A. D. Keedwell, *Latin Squares and Their Applications*, Akadémiai Kiadó, Budapest, 1974.

[19] W. Diffie and M. E. Hellman, "New Directions in Cryptography", IEEE Trans. on Info. Th., Vol. IT-22, pp. 644-654, Nov. 1976.

[20] W. Diffie and M. E. Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard", Computer, Vol. 10, pp. 74-78, June 1977.

[21] I.S.O. DP 10118, *Hash-functions for Digital Signatures*, I.S.O., April 1989.

[22] ISO/IEC CD 10118, *Information technology – Security techniques – Hash-functions*, I.S.O., 1991.

[23] J. H. Evertse, "Linear Structures in Block Ciphers", Advances in Cryptology – EUROCRYPT'87, Proceedings, LNCS 304, pp. 249-266, Springer-Verlag, 1988.

[24] P. Godlewski and C. Mitchell, "Key-Minimal Cryptosystems for Unconditional Secrecy", Journal of Cryptology, Vol. 3, No. 1, pp. 1-25, 1990.

[25] N. Guido and C. Harpes, "Sicherheitsanalysen eines neuen Blockverschlüsselungsverfahren", Semester Project, Swiss Federal Institute of Technology, July 1991.

[26] M. Iosifescu, *Finite Markov Processes and Their Applications*, John Wiley & Sons, 1980.

[27] H. N. Jendal, Y. J. Kuhn and J. L. Massey, "An Information - Theoretical Treatment of Homophonic Substitution", Advances in Cryptology - EUROCRYPT'89, Proceedings, LNCS 434, pp. 382-394, Springer-Verlag, 1990.

[28] J. B. Kam and G. I. Davida, "Structured Design of Substitution-Permutation Encryption Networks", IEEE Trans. on Computers, Vol. C-28, No. 10, pp. 747-753.

[29] J. Keilson, *Markov Chain Models - Rarity and Exponentiality*, Applied Mathematical Sciences, Vol. 28, Springer-Verlag, New York, 1979.

[30] A. G. Konheim, *Cryptography : A Primer*, New York: Wiley - Interscience, 1981.

[31] X. Lai and J. L. Massey, "A Proposal for a New Block Encryption Standard", Advances in Cryptology - EUROCRYPT'90, Proceedings, LNCS 473, pp. 389-404, Springer-Verlag, Berlin, 1991.

[32] X. Lai, J. L. Massey and S. Murphy, "Markov Ciphers and Differential Cryptanalysis", Advances in Cryptology - EUROCRYPT'91, Proceedings, LNCS 547, pp. 17-38, Springer-Verlag, Berlin, 1991.

[33] X. Lai and J. L. Massey, "Hash Functions based on Block Ciphers", to appear in Advances in Cryptology - EUROCRYPT'92, Proceedings, Springer-Verlag.

[34] J. L. Massey, "An Introduction to Contemporary Cryptology", *Proc. IEEE*, Vol. 76, No. 5, pp. 533-549, May 1988.

[35] J. L. Massey, "Cryptography — A Selective Survey", *Digital Communications* (Eds. E. Biglieri and G. Prati), Amsterdam: North-Holland, pp. 3-21, 1986. Reprinted as Invited Paper in *Alta Frequenza*, Vol. 55, No. 1, pp. 4-11, Jan.-Feb. 1986.

[36] J. L. Massey, *Cryptography, Fundamentals and Applications*, Copies of transparencies, Advanced Technology Seminars, 1988.

[37] J. L. Massey, U. Maurer and M. Wang, "Non-expanding, Key-minimal, Robustly-perfect, Linear and Bilinear Ciphers", Advances in Cryptology - EUROCRYPT'87, proceedings, LNCS 304, pp. 237-247, Springer-Verlag, 1988.

[38] S. M. Matyas, "Key Processing with Control Vectors", Journal of Cryptology, Vol. 3, No. 2, pp. 113–136, 1991.

[39] S. M. Matyas, C. H. Meyer and J. Oseas, "Generating Strong One-way Functions with Cryptographic Algorithm", IBM Technical Disclosure Bulletin, Vol. 27, No. 10A, pp. 5658-5659, March 1985.

[40] W. Meier, O. Staffelbach, " Nonlinearity Criteria for Cryptographic Functions", Advances in Cryptology – EUROCRYPT'89, Proceedings, LNCS 434, pp. 549-562, Springer-Verlag, 1990.

[41] R. C. Merkle, "One Way Hash Functions and DES", Advances in Cryptology – CRYPTO'89, Proceedings, LNCS 435, pp. 428-446, Springer-Verlag, 1990.

[42] C. H. Meyer and S. M. Matyas, *Cryptography: A New Dimension in Computer Data Security*, John Wiley & Sons, New York, 1982.

[43] C. H. Meyer and M. Schilling, "Secure Program Code with Modification Detection Code", Proceedings of SECURICOM 88, pp. 111-130, SEDEP.8, Rue de la Michodies, 75002, Paris, France.

[44] H. Minc, *Nonnegative Matrices*, John Wiley & Sons, New York, 1988.

[45] C. J. Mitchell, F. Piper and P. Wild, "Digital Signatures", *Contemporary Cryptology* (Ed. G. Simmons), pp. 325-378, IEEE Press, 1991.

[46] S Miyaguchi, K. Ohta and M. Iwata, "Confirmation that Some Hash Functions Are Not Collision Free", Advances in Cryptology – EUROCRYPT'90, Proceedings, LNCS 473, pp. 326-343, Springer-Verlag, Berlin, 1991.

[47] M. Naor and M. Yung, "Universal One-way Hash Functions and Their Cryptographic Applications", Proc. 21 Annual ACM Symposium on Theory of Computing, Seattle, Washington, May 15-17, 1989, pp. 33-43.

[48] B. Preneel, A. Bosselaers, R. Govaerts and J. Vandewalle, "Collision-free Hashfunctions Based on Blockcipher Algorithms." Proceedings of 1989 International Carnahan Conference on Security Technology, pp. 203-210.

[49] J. J. Quisquater and M. Girault, "2n-bit Hash Functions Using n-bit Symmetric Block Cipher Algorithms", Abstracts of EUROCRYPT'89.

[50] J. J. Quisquater and M. Girault, "2n-bit Hash Functions Using n-bit Symmetric Block Cipher Algorithms", Advances in Cryptology-EUROCRYPT'89, Proceedings, LNCS 434, pp. 102-109, Springer-Verlag, Berlin, 1990.

[51] R. A. Rueppel, *Analysis and Design of Stream Ciphers*, New York, NY, Springer-Verlag, 1986.

[52] C. E. Shannon, "The Synthesis of Two-Terminal Switching Circuits", Bell. System Technical Journal, Vol. 28, pp. 59-98, Oct. 1949.

[53] C. E. Shannon, "Communication Theory of Secrecy Systems", Bell. System Technical Journal, Vol. 28, pp. 656-715, Oct. 1949.

[54] A. Shimizu and S. Miyaguchi, "Fast Data Encryption Algorithm Feal", Advances in Cryptology - EUROCRYPT'87, Proceedings, LNCS 304, pp. 267-278, Springer-Verlag, 1988.

[55] W. Tuchman, "Hellman presents no shortcut solutions to the DES", IEEE Spectrum, Vol. 16, No. 7, p. 40, July 1979.

[56] G. S. Vernam, "Cipher Printing Telegraph Systems for Secure Wire and Radio Telegraphic Communications", J. Amer. Inst. Elec. Eng., Vol. 55, pp. 109-115, 1926.

[57] A. F. Webster, S. E. Tavares, "On the design of S-boxes", Advances in Cryptology - CRYPTO'85, LNCS 218, Proceedings, pp. 523-534, Springer-Verlag, 1986.

[58] I. Wegener, *The Complexity of Boolean Functions*, New York, NY: John Wiley & Sons, 1987.

[59] R. S. Winternitz, "Producing One-Way Hash Function from DES", Advances in Cryptology - CRYPTO'83, Proceedings, pp. 203-207, Plenum Press, New York, 1984.

[60] R. S. Winternitz, "A Secure One-way Hash Function Built from DES", Proc. 1984 IEEE Symposium on Security and Privacy, Oakland, 1984, pp. 88-90.

Index

Curriculum Vitae

Xuejia Lai, born on **June 4**, 1954, in Shanghai, China.

1960 - 1966 Primary school, Xian, China.

1966 - 1970 Middle school, Xian, China.

1970 - 1972 Building railway tunnel in Ankang, Shaanxi province, China.

1972 - 1977 Mechanic, Public Transportation Co., Xian, China.

1978 - 1982 Student, North-West Telecommunication Engineering Institute (now Xidian University), Xian, China. *Diplom El.Eng.* awarded in Jan. 1982. *B.Sc. El.Eng. degree* awarded in Apr. 1982.

1982 - 1984 Graduate student, Xidian University, Xian, China. *M.Sc. Math. degree* awarded in Nov. 1984.

1984 - 1985 Teaching assistant at Xidian University, Xian, China.

1985 - 1988 Graduate studies, Signal and Information Processing Laboratory, ETH, Zürich. *Nachdiplom in Nachrichtentechnik* awarded in Mar. 1988.

1988 - 1992 Doctoral Student, Signal and Information Processing Laboratory, ETH, Zürich.