Principles for Designing Secure Block Ciphers and One-Way Hash Functions

by

Yuliang Zheng

B.S., Computer Science, July 1982
Southeastern University, China

M.E., Electrical and Computer Engineering, March 1988
Yokohama National University

Submitted to
The Division of Electrical and Computer Engineering
In Partial Fulfillment of the Requirements for
The Degree of

DOCTOR OF PHILOSOPHY

In Electrical and Computer Engineering

YOKOHAMA NATIONAL UNIVERSITY

December 25, 1990

Principles for Designing Secure Block Ciphers and One-Way Hash Functions

by

Yuliang Zheng

Submitted to the
Division of Electrical and Computer Engineering
In Partial Fulfillment of the Requirements for the
Degree of

DOCTOR OF PHILOSOPHY

In Electrical and Computer Engineering

Signature of Author _____

Certified by

Professor	Hideki Imai (Thesis Supervisor)	_____
Professor	Yasunori Dohi	_____
Professor	Rokuya Ishii	_____
Associate Professor	Ryuji Kohno	_____
Associate Professor	Tsutomu Matsumoto	_____

YOKOHAMA NATIONAL UNIVERSITY

February 19, 1991

Acknowledgements

I am most grateful to Professor Hideki Imai, my thesis supervisor, for his continuous support, encouragement and guidance. I am also very grateful to Professor Tsutomu Matsumoto for much advice and constructive criticism. Thanks are also due to Professor Ryuji Kohno for helpful comments, to Professors Yasunori Dohi and Rokuya Ishii for serving on my thesis committee, and to Professor Kenichi Tamano for many suggestions.

I benefited a lot from stimulating discussions with many people, in particular with Feng Guo of Doshisha University, Kouichi Sakurai of Mitsubishi Electric Corporation (Melco), Atsushi Jun Fujioka, Kazuo Ohta and Tatsuaki Okamoto of Nippon Telegraph and Telephone Corporation (NTT), Toshiya Eleanor Itoh, Mitsunori Ogiwara and Osamu Watanabe of Tokyo Institute of Technology (Titech), and John Leo of Massachusetts Institute of Technology (MIT).

Finally, I am greatly indebted to my wife and best friend, Peipei. This thesis would be impossible without her understanding and patience.

Contents

Cover i

Non-digital Signatures ii

Acknowledgements iii

Contents iv

List of Figures vii

List of Tables viii

Abstract ix

I Designing Secure Block Ciphers 1

1 Preliminaries 2

1.1 Pseudo-random String/Function Generators 4

1.1.1 Pseudo-random String Generators 4

1.1.2 Pseudo-random Function Generators 6

1.2 Feistel-Type Transformation (FTT) 8

2 An Impossibility Result 10

2.1 $\psi(f^{(k)}, f^{(j)}, f^{(i)})$ Is Not Robust 11

2.2 Extensions 15

2.3 Remarks 16

3 Cryptographically Useful Transformations 18

3.1 Various Transformations 18

3.1.1 Type-1 Transformations 18

3.1.2 Type-2 Transformations 19

3.1.3 Type-3 Transformations 20
3.1.4 Generalized Transformations 20
3.2 Theorems on the Transformations 21
3.3 Proof for Theorem 5 23
3.4 Minimum Rounds for Robustness 26
3.5 Optimal Transformations 30
3.6 Notion of Super-Robustness 31

4 Block Ciphers Based on Type-2 Transformations 34
4.1 A Few Observations 34
4.2 PSBC — A Provably Secure Block Cipher 35
4.3 A Variant of PSBC 36
4.3.1 A Robust Pseudo-random String Generator 36
4.3.2 PSBC with Key-Expanding 39
4.4 Practical Block Ciphers 40

II Constructing One-Way Hash Functions 43

5 Definitions 44
5.1 One-Way Functions 44
5.2 Universal Hash Functions 47
5.3 One-Way Hash Functions 48
5.3.1 Universal One-Way Hash Functions 48
5.3.2 Collision Intractable Hash Functions 51

6 Duality between Two Cryptographic Primitives 53
6.1 Extension and Compression Methods 53
6.1.1 Serial Versions 53
6.1.2 Parallel Versions 56
6.2 PSGs and UOHs from One-Way Permutations 56
6.2.1 PSGs from One-Way Permutations 56
6.2.2 UOHs from One-Way Permutations 57
6.3 Practical One-Way Hash Functions 61
6.3.1 Compressing via the Rabin Encryption Function 63
6.3.2 Compressing via xDES 63

7 Structural Properties of One-Way Hash Functions **66**
7.1 Transforming UOH/U into UOH/$EN[\ell]$ 66
7.2 UOHs from One-way Quasi-Injections 69
7.2.1 One-Way Quasi-Injections 69
7.2.2 The Construction 70
7.3 A Hierarchy of One-Way Hash Functions 72

8 Conclusion **78**

Bibliography **80**

Vita **84**

List of Figures

1.1 A Secure Communication System 2
1.2 Feistel-Type Transformation (FTT) 8
1.3 Feistel-Type Transformation (FTT) Lemma 9

2.1 Structure of Oracle Circuit T_{2n} 12
2.2 Type-1 Transformation 15
2.3 Structure of Oracle Circuit $T_{\ell n}$ 17

3.1 Type-2 Transformation 19
3.2 Type-3 Transformation 20
3.3 Inverse of Type-3 Transformation 21

4.1 Enciphering Algorithm for PSBC 36
4.2 Algorithm G_n for Computing S_{2n2^n} 38
4.3 Enciphering Algorithm for PSBC with Key-Expanding 40

5.1 One-Way Hash Functions as Anti-Virus 45
5.2 UOH as a Three-Party Game 49

6.1 Serial-Extension 1 54
6.2 Serial-Extension 2 55
6.3 Serial-Compression 1 56
6.4 Serial-Compression 2 57
6.5 PSG from a One-Way Permutation 58
6.6 UOH/$EN[\ell]$ from a One-Way Permutation 59
6.7 Practical Hashing Method 62

7.1 UOH/$PSE[\ell]$ is Strictly Included in UOH/U 75
7.2 UOH/PSE[poly] versus CIH 76
7.3 Hierarchical Structure of One-Way Hash Functions 77

List of Tables

3.1 Summary of Theorems 5 – 7 22
3.2 Minimum Rounds λ for Robustness 30

4.1 Four Example Ciphers 42

5.1 Classification of UOH 51
5.2 Classification of CIH 52

6.1 Two Practical One-Way Hash Functions 65

Abstract

This thesis is concerned with issues of designing secure (secret-key) block ciphers and constructing one-way hash functions. Both block ciphers and one-way hash functions are indispensable to secure information systems built on cryptographic techniques. With a block cipher, we can safeguard our important information transmitted over insecure communication networks. And with a one-way hash function, we can safely compress very long messages into relatively short ones to improve the overall efficiency of an information system or to detect un-authorized modifications to these messages.

The thesis consists of two parts. Part I deals with designing secure block ciphers and Part II with constructing one-way hash functions. The outlines of the two parts are as follows.

In Part I, we first prove an impossibility result on constructing pseudo-random permutations from random functions, which is closely related to the design of secure block ciphers. Then we consider the problem of constructing block ciphers which have the following ideal properties:

1. The ciphers are provably secure,
2. Security of the ciphers does not depend on any unproved hypotheses,
3. The ciphers can be easily implemented with current technology, and
4. All design criteria for the ciphers are made public.

It is currently unclear whether or not there really exists such an ideal block cipher. So to meet the requirements of practical applications, the best thing we can do is to construct a block cipher such that it approximates the ideal one as closely as possible. In this thesis, we make a significant step in this direction. In particular, we construct several block ciphers which have the above mentioned properties 2, 3 and 4 as well as the following one:

1′. Security of the ciphers is supported by convincing evidence.

In Part II, we first reveal a duality between constructions of two basic cryptographic primitives, *pseudo-random string generators* and *one-way hash functions*. Applying the duality, we present a construction for *universal one-way hash functions* assuming the existence of one-way permutations. Using ideas behind the construction, we propose *practical* one-way hash functions, the fastest of which compress nearly $2n$-bit long input into n-bit long output strings by applying only *twice* a one-way function.

Then we prove that universal one-way hash functions with respect to initial-strings chosen *arbitrarily* exist if and only if universal one-way hash functions with respect to initial-strings chosen *uniformly at random* exist. As an application of the result, we show that universal one-way hash functions with respect to initial-strings chosen arbitrarily can be constructed under the assumption of the existence of one-way *quasi*-injections.

Finally, we investigate relationships among various versions of one-way hash functions. We prove that some versions of one-way hash functions are strictly included in others by explicitly constructing hash functions that are one-way in the sense of the former but not in the sense of the latter.

Most of the results contained in this thesis were announced at EuroCrypt'89 [ZMI89a], Crypto'89 [ZMI89b], AAECC-8 [ZMI90a], SCIS90 [ZMI90b], and Crypto'90 [ZMI90c].

Part I

Designing Secure Block Ciphers

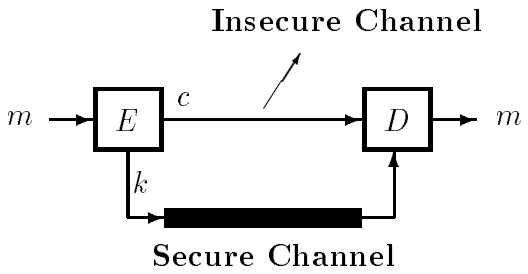
Chapter 1

Preliminaries

Secret-key block ciphers (or shortly, block ciphers) are widely applied in secure information systems. Figure 1.1 shows a typical communication system using a block cipher. About *fifteen* years ago, IBM designed a block cipher called Data Encryption Standard (DES) [NBS77]. DES, the first modern block cipher whose algorithm is publicly available, is a kind of product ciphers with Lucifer as its direct predecessor [FNS75, Kon81]. A little more specifically, both DES and Lucifer consist of 16 rounds of *Feistel-type transformations (FTT's)* which are invented by and named (by us) after Feistel.

From the beginning of DES, however, there had a lot of controversy about its security, and especially, about its design criteria [Kon81] which have been classified by NSA and its designer IBM. Many computer scientists and cryptography experts were concerned about the possibilities that DES may possess weaknesses only NSA and IBM are aware of, and that trap-doors may have been inserted into the S-boxes of DES which would give a cryptoanalytic advantage to a knowledgeable party. For these reasons, a great amount of effort has been invested in attempting to break the cipher, or to find its weaknesses. And many researchers have tried revealing the myths around the design criteria.

In their nice paper [LR88], Luby and Rackoff showed that DES would be provably secure if its f-functions were secure pseudo-random ones. Unfortunately, the f-functions of DES cannot be secure in any reasonable sense. In the same paper, Luby and Rackoff proved also a result about FTT's: A function consisting of three rounds of randomly and independently chosen FTT's, which is in fact a permutation, cannot be efficiently distinguished from a truly random one. This result is very appealing, since it relies on no unproved hypotheses, and more importantly, it suggests that there is an extremely *simple* constructive method for designing a theoretically secure block cipher which does not rely on any unproved hypotheses. However, it



E — Encryption Algorithm
D — Decryption Algorithm
m — Plain Text
c — Cipher Text
k — Key
$c = E(k, m)$
$m = D(k, c)$

Figure 1.1: A Secure Communication System

is practically impossible to construct such a cipher, simply because it takes a huge amount of memory to implement the cipher.

Therefore both practical needs and theoretical interest encourage us to seek for an ideal block cipher having the following properties:

1. The cipher is provably secure,
2. Security of the cipher does not depend on any unproved hypotheses,
3. The cipher can be easily implemented with current technology, and
4. All design criteria for the cipher are made public.

It is still an open problem whether or not there really exists such a block cipher. The best thing we can do currently is to construct a block cipher such that it approximates the ideal one as closely as possible.

In Part I, we will propose an interesting kind of transformations — *Generalized Type-2 transformations*, and show that it is an excellent building block for cryptosystems. Utilizing this type of transformations, we construct several concrete block ciphers which have the above mentioned properties (2), (3) and (4) as well as the following one:

1. Security of the cipher is supported by convincing evidence.

Our results build upon profound mathematical bases for information security recently established in a series of excellent papers such as [BM84, Yao82, Lev87, GGM86, Sch88], [1] and especially [LR88].

Part I is organized as follows: In the remaining portion of this chapter, we define terminology used later, review one of the main design rules for DES — FTT's, and introduce the result of Luby and Rackoff on the rule. In Chapter 2, we prove an impossibility result on constructing pseudo-random permutations from random functions, which is closely related to the design of secure block ciphers. Chapter 3 proposes various types of transformations and shows that all these transformations can be used to construct permutations not efficiently distinguishable from a truly random one. Among the transformations, Generalized Type-2 ones are proved to be most preferable. Chapter 4 constructs a theoretically provably secure block cipher (PSBC) by the use of Generalized Type-2 transformations, and also presents a variant of PSBC. Several concrete block ciphers are shown at the end of in the same chapter.

1.1 Pseudo-random String/Function Generators

For purposes which will become clear later, our notions introduced below are slight generalizations of those given in [Yao82, GGM86, LR88], mainly in the following aspect: In contrast to those in [Yao82, GGM86, LR88], we will not impose polynomial bound upon the running time of an algorithm realizing a pseudo-random string/function generator or on the size of a (local) statistical test for strings/functions.

1.1.1 Pseudo-random String Generators

The set of positive integers is denoted by $\mathcal{N}$. Let $\Sigma = \{0,1\}$ be the alphabet we consider. For $n \in \mathcal{N}$, denote by Σ^n the set of all strings over Σ with length n. The concatenation of two strings x, y is denoted by $x \diamond y$, and if no confusion arises, it is denoted by xy. For $s_1, s_2 \in \Sigma^n$, let $s_1 \oplus s_2$ denote the bit-wise XOR of the two strings. The *composition* of two functions f and g, denoted by $f \circ g$, is defined by $f \circ g(x) = f(g(x))$ for all $x \in \Sigma^n$. And in particular, $f \circ f$ is denoted by $f^{(2)}$, $f \circ f \circ f$ by $f^{(3)}$, and so on. The number of elements in a finite set X is denoted by $\sharp X$. By $x \in_R X$ we mean that x is drawn randomly and uniformly from the set X.

[1] In Chapter 2, we will show that the main result of [Sch88] was false. But in this thesis the correct version of the result is used.

By a function in n (or t etc.) we mean, unless otherwise specified, a function from $\mathcal{N}$ to $\mathcal{N}$. Now let P be a function in n with $P(n) > n$. A *pseudo-random string generator* (PSG) is a collection of functions $S = \{S_n \mid n \in \mathcal{N}\}$, where each function S_n maps an n-bit string *seed* into a $P(n)$-bit string $S_n(seed)$ and it can be computed by some deterministic algorithm.

Security (or strength) of PSG's is defined in terms of local statistical tests for strings.

Definition 1 *Let Θ and $\mathcal{L}$ be sets of functions in n, and Υ a set of functions from $\mathcal{N}$ to $[0,1]$. Let P be a function in n with $P(n) > n$, and let $\theta \in \Theta$ and $L \in \mathcal{L}$ with $0 < L(n) \leq P(n)$. A family of circuits $T^s = \{T^s_n \mid n \in \mathcal{N}\}$ is called a* local (θ, L) statistical test for strings *if each T^s_n is of size $\theta(n)$,* [2] *and on input an $L(n)$-bit fixed portion of a $P(n)$-bit string x, outputs a single bit $T^s_n[x]$. Call θ the size of T^s. Now let $S = \{S_n \mid n \in \mathcal{N}\}$ be a PSG where S_n maps an n-bit string into a $P(n)$-bit one. We say that*

1. *S* locally ε-passes the test *T^s if for all sufficiently large n,*

$$|\Pr\{T^s_n[r] = 1\} - \Pr\{T^s_n[S_n(seed)] = 1\}| < \varepsilon(n),$$

   where $r \in_R \Sigma^{P(n)}$, $seed \in_R \Sigma^n$ and $\varepsilon \in \Upsilon$;

2. *S is* locally (θ, L, ε)-secure *if it locally ε-passes* all (θ, L) *tests;*

3. *S is* locally $(\Theta, \mathcal{L}, \Upsilon)$-secure *if it is locally (θ, L, ε)-secure for any $\varepsilon \in \Upsilon$ and any $(\theta, L) \in \Theta \times \mathcal{L}$ with $0 < L(n) \leq P(n)$.*

Especially, a locally $(\Theta, \mathcal{L}, \Upsilon)$-secure PSG S is said

4. locally $(\infty, \mathcal{L}, \Upsilon)$-secure *if Θ is the set of all functions in n, and*

5. robust *if, furthermore, $\mathcal{L}$ is the infinite set of all polynomials in n and Υ that of the reciprocals of all polynomials in n. (The reciprocal of a polynomial Q in n is defined by $1/Q(n)$.)*

Finally, assume that $S = \{S_n \mid n \in \mathcal{N}\}$ is a PSG where S_n can be computed in deterministic polynomial *time in n. Then*

6. *S is called* locally polynomially secure *if it is locally $(\Theta, \mathcal{L}, \Upsilon)$-secure where both Θ and $\mathcal{L}$ are the infinite set of all polynomials in n, and Υ that of the reciprocals of all polynomials in n.*

[2] The size of a circuit is the total number of connections in the circuit.

Note that Yao's definition for *polynomial size statistical tests for strings* [Yao82, GGM86] is obtained from ours by letting P, θ and L be polynomials in n with $P = L$. Now assume, as at the end of Definition 1, that $S = \{S_n \mid n \in \mathcal{N}\}$ is a PSG where S_n can be computed in deterministic *polynomial* time in n. For such a PSG S, Yao defined that it *passes a polynomial size statistical test for strings* $T^s = \{T^s_n \mid n \in \mathcal{N}\}$ if for any polynomial P_1 and for all sufficiently large n,

$$|\Pr\{T^s_n[r] = 1\} - \Pr\{T^s_n[S_n(seed)] = 1\}| < 1/P_1(n),$$

where $r \in_R \Sigma^{P(n)}$ and $seed \in_R \Sigma^n$, and that S is *polynomially secure* if it passes *all* polynomial size statistical tests for strings.

Fact 1 *Assume that $S = \{S_n \mid n \in \mathcal{N}\}$ is a PSG where S_n can be computed in deterministic* polynomial *time in n. Then S is polynomially secure iff it is locally polynomially secure.*

Proof : It is an immediate consequence of Yao's famous theorem on statistical tests for strings [Yao82, GGM86]. Here we give a direct proof for it. Let P be a polynomial, and suppose that S_n maps an n-bit string into a $P(n)$-bit one. The "if" part is clearly true. To prove the "only if" part, it suffices to show that if S passes all polynomial size statistical tests for strings then for any polynomials L, P_3 and P_4 with $0 < L(n) < P(n)$, it also locally $1/P_4$-passes all local (P_3, L) statistical test for strings. Assume for contradiction that S does not $1/P_4$-passes a local (P_3, L) statistical test T^s for strings. Then one can easily construct from T^s a polynomial size statistical test for strings $U^s = \{U^s_n \mid n \in \mathcal{N}\}$ such that the size of U^s is at most $P \cdot P_3$ and that S does not pass it. □

1.1.2 Pseudo-random Function Generators

Let P be an increasing function in n. A *pseudo-random function generator* (PFG) is a collection of functions $F = \{F_n \mid n \in \mathcal{N}\}$, where F_n specifies for each $P(n)$-bit string key, (the description of) a function $F_n(key) \in \mathcal{H}_n$ that can be computed by some deterministic algorithm, where $\mathcal{H}_n$ denotes the set of all 2^{n2^n} functions from Σ^n to Σ^n.

Security of a PFG is defined in terms of statistical tests for functions, and the latter uses the concept of oracle circuits which are counterparts of often used *oracle Turing machines*. An *oracle circuit* C_n is an acyclic circuit which contains, in addition to ordinary AND, OR, NOT and constant gates, also a particular kind of gates —

oracle gates. Each oracle gate has an n-bit input and an n-bit output, and it is evaluated using some function from $\mathcal{H}_n$. The output of C_n, a single bit, is denoted by $C_n[f]$ when a function $f \in \mathcal{H}_n$ is used to evaluate the oracle gates.

Definition 2 *Let Θ and $\mathcal{Q}$ be sets of functions in n, and Υ a set of functions from $\mathcal{N}$ to $[0,1]$. Let $\theta \in \Theta$ and $Q \in \mathcal{Q}$ be two functions with $0 \leq Q(n) < \theta(n)$. A family of circuits $T^f = \{T_n^f \mid n \in \mathcal{N}\}$ is called a* (θ, Q) statistical test for functions *where T_n^f is an oracle circuit which is of size $\theta(n)$ and has $Q(n)$ oracle gates. Let P be an increasing function in n, and $F = \{F_n \mid n \in \mathcal{N}\}$ a PFG where F_n specifies for each $P(n)$-bit string key a function $F_n(key) \in \mathcal{H}_n$. We say that*

1. *F* ε-passes the test *T^f if for all sufficiently large n,*

$$|\Pr\{T_n^f[r] = 1\} - \Pr\{T_n^f[F_n(key)] = 1\}| < \varepsilon(n)$$

   where $r \in_R \mathcal{H}_n$, $key \in_R \Sigma^{P(n)}$ and $\varepsilon \in \Upsilon$.

2. *F is* (θ, Q, ε)-secure *if it ε-passes* all (θ, Q) *tests.*

3. *F is* $(\Theta, \mathcal{Q}, \Upsilon)$-secure *if it is (θ, Q, ε)-secure for any $\varepsilon \in \Upsilon$ and any $(\theta, Q) \in \Theta \times \mathcal{Q}$ with $0 \leq Q(n) < \theta(n)$.*

Especially,

4. *a $(\Theta, \mathcal{Q}, \Upsilon)$-secure PFG F is said* $(\infty, \mathcal{Q}, \Upsilon)$-secure *when Θ is the set of all functions in n.*

5. *an $(\infty, \mathcal{Q}, \Upsilon)$-secure PFG F is said* robust *if $\mathcal{Q}$ is the infinite set of all polynomials in n and Υ that of the reciprocals of all polynomials in n.*

Finally assume that for each n and for each key $\in \Sigma^{P(n)}$, the function $F_n(key)$ can be computed in deterministic polynomial *time in n. (This implies that P is a polynomial in n.) Then*

6. *F is called* polynomially secure *if it is $(\Theta, \mathcal{Q}, \Upsilon)$-secure, where both Θ and $\mathcal{Q}$ are the infinite set of all polynomials in n and Υ that of the reciprocals of all polynomials in n.*

We are mainly interested in a special kind of PFG's — pseudo-random permutation generators which are invertible. Let P be an increasing function in n. A *pseudo-random permutation generator* is a pseudo-random function generator $F = \{F_n \mid n \in \mathcal{N}\}$, where F_n specifies for each $P(n)$-bit string *key* a *permutation* $F_n(key) \in \mathcal{S}_n$ that

can be computed by some deterministic algorithm, where $\mathcal{S}_n$ denotes the set of all $2^n!$ permutations on Σ^n. A pseudo-random permutation generator $F = \{F_n \mid n \in \mathcal{N}\}$ is called *invertible* if there is a pseudo-random permutation generator

$$\tilde{F} = \{\tilde{F}_n \mid n \in \mathcal{N}\}$$

such that for each $P(n)$-bit string key, $\tilde{F}_n$ specifies the inverse of $F_n(key)$. Security of (invertible) pseudo-random permutation generators is defined in exactly the same way as for pseudo-random function generators.

1.2 Feistel-Type Transformation (FTT)

For a function $f_i \in \mathcal{H}_n$, we associate with it a function $g_i \in \mathcal{H}_{2n}$ defined by

$$g_i(B_1, B_2) = (B_2 \oplus f_i(B_1), B_1)$$

where $B_1, B_2 \in \Sigma^n$ and $\mathcal{H}_{2n}$ is the set of all functions from Σ^{2n} to Σ^{2n} (Figure 1.2). Note that g_i is obtained from f_i by applying one of the main design rules for DES, and it corresponds roughly to a layer of DES. Since the design rule was due to Feistel, we call g_i a *Feistel-type transformation* (FTT). Note that our definition for g_i is *notationally* different from that given in [LR88, Sch88]. However, the difference is *not* essential, and does *not* affect the results to be proved.

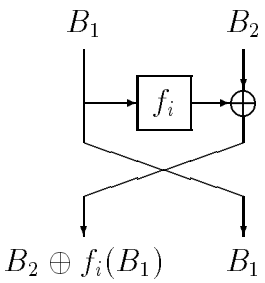


Figure 1.2: Feistel-Type Transformation (FTT)

For $f_1, f_2, \ldots, f_s \in \mathcal{H}_n$, let

$$\psi(f_s, \ldots, f_2, f_1) = g_s \circ \cdots \circ g_2 \circ g_1.$$

We say that $\psi(f_s, \ldots, f_2, f_1)$ consists of s *rounds* of FTT's. Obviously, g_i is an invertible permutation, and hence so is $\psi(f_s, \ldots, f_2, f_1)$.

Luby and Rackoff proved the following result which was called *Main Lemma* in [LR88] but is called *FTT Lemma* in this paper: For independent random functions

$f_1, f_2, f_3 \in_R \mathcal{H}_n$, it is infeasible to distinguish $\psi(f_3, f_2, f_1)$ from a function drawn randomly and uniformly from $\mathcal{H}_{2n}$. (See Figure 1.3.)

Lemma 1 (FTT Lemma, Version 1, [LR88]) *Let Q be a polynomial in n and C_{2n} be an oracle circuit with $Q(n) < 2^n$ oracle gates. Then*

$$|\Pr\{C_{2n}[r] = 1\} - \Pr\{C_{2n}[\psi(f_3, f_2, f_1)] = 1\}| \leq \frac{Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{2n}$ and $f_1, f_2, f_3 \in_R \mathcal{H}_n$.

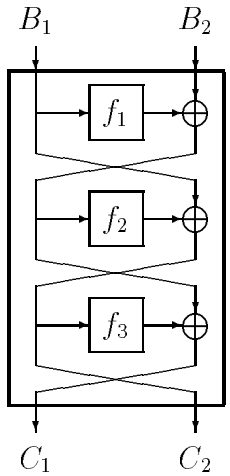


Figure 1.3: Feistel-Type Transformation (FTT) Lemma

FTT Lemma is surprising in the sense that it does not depend on any unproved hypotheses. It implies that we can construct as follows a block cipher which does not relying on any assumption and is provably secure against chosen-plaintext attack: Let the length of a plaintext be $2n$. Choose randomly and uniformly from $\mathcal{H}_n$ three functions f_1, f_2 and f_3, and let the enciphering algorithm be $\psi(f_3, f_2, f_1)$ and the deciphering algorithm be the inverse of $\psi(f_3, f_2, f_1)$.

However one soon finds that such an approach is impractical: To make the cipher secure against some trivial attacks such as exhaustive search, $2n$ must be sufficiently large, say ≥ 64, i.e., $n \geq 32$. When $n = 32$, specifying $\psi(f_3, f_2, f_1)$ takes at least $3 \cdot 32 \cdot 2^{32} \approx 4 \cdot 10^{11}$ bits, which is infeasible currently and even in the foreseeable future. In other words, there is still a big gap between practically constructing a provably secure block cipher and the nice theory initiated by Luby and Rackoff. In Chapter 3 we will examine various types of transformations, and fill the gap greatly. But before proceeding to the issue, we first prove an impossibility result related to FTT Lemma.

Chapter 2

An Impossibility Result

Let $f_i \in \mathcal{H}_n$. Recall the Feistel-type transformation associated with f_i is defined by $g_i(L,R) = (R \oplus f_i(L), L)$, where $L, R \in \Sigma^n$. For $f_1, f_2, \ldots, f_s \in \mathcal{H}_n$, define $\psi(f_s, \ldots, f_2, f_1) = g_s \circ \cdots \circ g_2 \circ g_1$. As mentioned in Chapter 1, Luby and Rackoff showed that from *three* independent random functions $f, g, h \in \mathcal{H}_n$, one can construct, by three applications of Feistel-type transformations, a robust permutation in $\mathcal{H}_{2n}$ which cannot be distinguished from a truly random function in $\mathcal{H}_{2n}$ by statistical tests for functions with polynomially many oracle gates [LR88].

A natural question is whether or not one can reduce the number of independent random functions from three to two or to one. In particular, one would like to ask whether or not permutations like $\psi(g, f, f)$ or $\psi(f, f, f^{(2)})$ are robust. In his Master Thesis [Oh88], Ohnishi observed that *two* independent random functions are sufficient in Luby and Rackoff's construction. In particular, he showed that both $\psi(g, f, f)$ and $\psi(g, g, f)$, where $f, g \in_R \mathcal{H}_n$, are robust.

In the same thesis, Ohnishi also showed that neither $\psi(f, f, f)$ nor $\psi(f, g, f)$ are robust. (This result was independently obtained by Rueppel in [Rue89].) [1] In fact, Ohnishi proved a stronger result — *both* $\psi(f_s, \ldots, f_2, f_1, f_0, f_1, f_2, \ldots, f_s)$ *and* $\psi(f_s, \ldots, f_2, f_1, f_1, f_2, \ldots, f_s)$, *where* $f_i \in \mathcal{H}_n$, *can be distinguished from an* $r \in_R \mathcal{H}_{2n}$ *by an oracle circuit* T_{2n} *with two oracle gates* O_1 *and* O_2. The structure of T_{2n} is as follows:

1. Choose $X_1, X_2 \in I_n$.

2. Input (X_1, X_2) to O_1. Assume that the output of O_1 is (Y_1, Y_2).

3. Input (Y_2, Y_1) to O_2. Assume that the output of O_2 is (Z_1, Z_2).

[1] At Eurocrypt'88, Schnorr [Sch88] *erroneously* claimed that $\psi(f, f, f)$, where $f \in_R \mathcal{H}_n$, cannot be efficiently distinguished from an $r \in_R \mathcal{H}_{2n}$.

4. T_{2n} outputs a bit 1 iff $(X_1, X_2) = (Z_2, Z_1)$.

However, it still remains open whether or not permutations like $\psi(f, f, f^{(2)})$ and $\psi(f^{(2)}, f, f)$ are robust. Ohnishi's technique as described above is not applicable to these cases.

We show in the next section that for any f and for any $i, j, k \in \mathcal{N}$, there is a circuit that distinguishes $\psi(f^{(k)}, f^{(j)}, f^{(i)})$ from an $r \in_R \mathcal{H}_{2n}$. Thus, as immediate consequences, we have that

1. none of $\psi(f, f, f)$, $\psi(f, f, f^{(2)})$ and $\psi(f^{(2)}, f, f)$ are robust and,
2. Ohnishi's constructions $\psi(g, f, f)$ and $\psi(g, g, f)$ are optimal among the robust permutations $\psi(f_3^{(k)}, f_2^{(j)}, f_1^{(i)})$ where $i, j, k \in \mathcal{N}$ and $f_1, f_2, f_3 \in \mathcal{H}_n$ such that for any $1 \leq s, t \leq 3$, either $f_s = f_t$ or f_s is independent of f_t.

In Section 2.2 we investigate extensions of our main result.

2.1 $\psi(f^{(k)}, f^{(j)}, f^{(i)})$ Is Not Robust

Assume that Q_1 is a polynomial and that $T^f = \{T_n^f \mid n \in \mathcal{N}\}$ is a statistical test for functions such that for each $n \in \mathcal{N}$, T_n^f has $Q_1(n)$ oracle gates. Let P be an increasing function in n, and $F = \{F_n \mid n \in \mathcal{N}\}$ a PFG where F_n specifies for each $P(n)$-bit string key a function $F_n(key) \in \mathcal{H}_n$. We say that T^f is a *distinguisher* for F if there is a polynomial Q_2 such that for infinitely many $n \in \mathcal{N}$,

$$|\Pr\{T_n^f[r] = 1\} - \Pr\{T_n^f[F_n(key)] = 1\}| \geq Q_2(n)$$

where $r \in_R \mathcal{H}_n$, $key \in_R \Sigma^{P(n)}$. Note that by definition, F is robust iff there is no distinguisher for F. It is convenient to say that $F_n(key)$, where $key \in_R \Sigma^{P(n)}$, is robust whenever F is robust, and not robust (or can be distinguished from a random function) otherwise.

Now we prove the main result on permutations $\psi(f^{(k)}, f^{(j)}, f^{(i)})$. For $i, j, k \in \mathcal{N}$, let $\Psi(i, j, k) = \{\Psi_{2n}(i, j, k) \mid n \in \mathcal{N}\}$ be a PFG, where for each $n \in \mathcal{N}$, $\Psi_{2n}(i, j, k)$ is a function that on input the description of $f \in \mathcal{H}_n$ outputs the description of $\psi(f^{(k)}, f^{(j)}, f^{(i)}) \in \mathcal{H}_{2n}$.

Theorem 1 *For any $i, j, k \in \mathcal{N}$, there is a distinguisher $T = \{T_{2n} \mid n \in \mathcal{N}\}$ for $\Psi(i, j, k)$, i.e., $\Psi(i, j, k)$ is not robust. Each T_{2n} has $(m_1 + m_2 + 1)$ oracle gates, where $m_1 = (i + j)/d$, $m_2 = (j + k)/d$ and $d = \gcd(i + j, j + k)$.*

Proof : Denote by $O_0, O_1, \ldots, O_{m_1+m_2}$ the (m_1+m_2+1) oracle gates, by (X_{s1}, X_{s2}) and (Y_{s1}, Y_{s2}) the input to and output of O_s respectively, and by 0^n the all-0 string in Σ^n. The structure of T_{2n} is as follows. See also Figure 2.1.

DESCRIPTION OF T_{2n}:

1. The input to O_0 is $(X_{01}, X_{02}) = (0^n, 0^n)$.
2. The input to O_1 is $(X_{11}, X_{12}) = (0^n, Y_{01})$. And if $m_1 > 1$, then for each $1 < p \leq m_1$, the input to O_p is $(X_{p1}, X_{p2}) = (0^n, X_{(p-1)2} \oplus Y_{(p-1)1})$.
3. The input to O_{m_1+1} is $(X_{(m_1+1)1}, X_{(m_1+1)2}) = (Y_{02}, 0^n)$. And if $m_2 > 1$, then for each $m_1+1 < t \leq m_1+m_2$, the input to O_t is $(X_{t1}, X_{t2}) = (X_{(t-1)1} \oplus Y_{(t-1)2}, 0^n)$.
4. Finally, T_{2n} outputs a bit 1 iff $Y_{m_1 2} = X_{(m_1+m_2)1} \oplus Y_{(m_1+m_2)2}$.

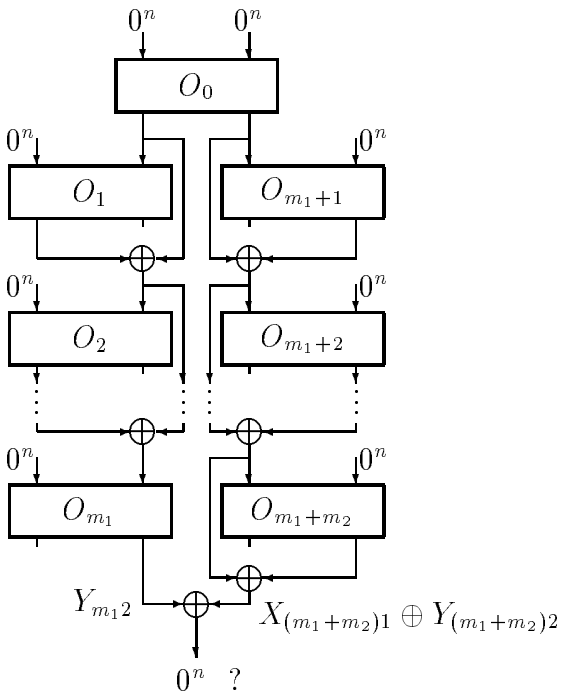


Figure 2.1: Structure of Oracle Circuit T_{2n}

Note that the number of oracle gates in T_{2n} is $m_1 + m_2 + 1$, a constant. Now we analyze the behavior of T_{2n} in the following two cases: CASE-1, where a function $\psi(f^{(k)}, f^{(j)}, f^{(i)}) \in \Psi_{2n}(i, j, k)$ is used to evaluate the oracle gates, and CASE-2, where a function r drawn randomly and uniformly from $\mathcal{H}_{2n}$ is used to evaluate the oracle

gates. We show that in the former case, the probability that T_{2n} outputs a bit 1 is 1 and, in the latter case, the probability is less than $1/2^{n-1}$. Thus $T = \{T_{2n} \mid n \in \mathcal{N}\}$ is a distinguisher for $\Psi(i,j,k)$.

CASE-1: Notice that $\psi(f^{(k)}, f^{(j)}, f^{(i)})(L,R) = (R \oplus f^{(i)}(L) \oplus f^{(k)}(L \oplus f^{(j)}(R \oplus f^{(i)}(L))), L \oplus f^{(j)}(R \oplus f^{(i)}(L)))$. Thus the output of O_0 is $(Y_{01}, Y_{02}) = (f^{(i)}(0^n) \oplus f^{(k+j+i)}(0^n), f^{(j+i)}(0^n))$.

Denote by $\sim$ a string which we do not care. The inputs to and outputs of $O_1, O_2, \ldots, O_{m_1}$ are as follows:

$$\begin{aligned}
O_1 : (X_{11}, X_{12}) &= (0^n, f^{(i)}(0^n) \oplus f^{(k+j+i)}(0^n)), \\
(Y_{11}, Y_{12}) &= (f^{(k+j+i)}(0^n) \oplus f^{(2k+2j+i)}(0^n), \sim); \\
O_2 : (X_{21}, X_{22}) &= (0^n, f^{(i)}(0^n) \oplus f^{(2k+2j+i)}(0^n)), \\
(Y_{21}, Y_{22}) &= (f^{(2k+2j+i)}(0^n) \oplus f^{(3k+3j+i)}(0^n), \sim); \\
&\cdots \\
O_{m_1} : (X_{m_1 1}, X_{m_1 2}) &= (0^n, f^{(i)}(0^n) \oplus f^{(m_1 k + m_1 j + i)}(0^n)), \\
(Y_{m_1 1}, Y_{m_1 2}) &= (\sim, f^{(m_1 k + (m_1+1) j + i)}(0^n)).
\end{aligned}$$

Similarly, for $O_{m_1+1}, O_{m_1+2}, \ldots, O_{m_1+m_2}$, we have:

$$\begin{aligned}
O_{m_1+1} : (X_{(m_1+1)1}, X_{(m_1+1)2}) &= (f^{(j+i)}(0^n), 0^n), \\
(Y_{(m_1+1)1}, Y_{(m_1+1)2}) &= (\sim, f^{(j+i)}(0^n) \oplus f^{(2j+2i)}(0^n)); \\
O_{m_1+2} : (X_{(m_1+2)1}, X_{(m_1+2)2}) &= (f^{(2j+2i)}(0^n), 0^n), \\
(Y_{(m_1+2)1}, Y_{(m_1+2)2}) &= (\sim, f^{(2j+2i)}(0^n) \oplus f^{(3j+3i)}(0^n)); \\
&\cdots \\
O_{m_1+m_2} : (X_{(m_1+m_2)1}, X_{(m_1+m_2)2}) &= (f^{(m_2 j + m_2 i)}(0^n), 0^n), \\
(Y_{(m_1+m_2)1}, Y_{(m_1+m_2)2}) &= (\sim, f^{(m_2 j + m_2 i)}(0^n) \oplus f^{((m_2+1) j + (m_2+1) i)}(0^n)).
\end{aligned}$$

Thus

$$\begin{aligned}
Y_{m_1 2} &= f^{(m_1 k + (m_1+1) j + i)}(0^n) = f^{(m_1 (k+j) + j + i)}(0^n) \\
&= f^{(\frac{i+j}{d}(k+j)+j+i)}(0^n) = f^{(j\frac{k+j}{d} + i\frac{k+j}{d} + j + i)}(0^n) \\
&= f^{(j m_2 + i m_2 + j + i)}(0^n) = f^{((m_2+1) j + (m_2+1) i)}(0^n) \\
&= X_{(m_1+m_2)1} \oplus Y_{(m_1+m_2)2},
\end{aligned}$$

and the probability that T_{2n} outputs a bit 1 is 1.

CASE-2: There are two sub-cases to be analyzed: $(Y_{01}, Y_{02}) = (0^n, 0^n)$ and $(Y_{01}, Y_{02}) \neq (0^n, 0^n)$.

1. When $(Y_{01}, Y_{02}) = (0^n, 0^n)$, we have $Y_{m_1 2} = X_{(m_1+m_2)1} \oplus Y_{(m_1+m_2)2}$. However the probability that (Y_{01}, Y_{02}) coincides with $(0^n, 0^n)$ is $1/2^{2n}$.

2. When $(Y_{01}, Y_{02}) \neq (0^n, 0^n)$, we have $(X_{11}, X_{12}) \neq (X_{(m_1+1)1}, X_{(m_1+1)2})$, and hence $(X_{11}, X_{12}) \neq (0^n, 0^n)$ or $(X_{(m_1+1)1}, X_{(m_1+1)2}) \neq (0^n, 0^n)$. Suppose that $(X_{11}, X_{12}) \neq (0^n, 0^n)$. (The other case is similar.) Then (Y_{11}, Y_{12}), and hence (Y_{21}, Y_{22}), (Y_{31}, Y_{32}), ..., $(Y_{m_1 1}, Y_{m_1 2})$ are all random strings in Σ^{2n}. These strings are independent of $(X_{(m_1+1)1}, X_{(m_1+1)2})$, and hence of $(Y_{(m_1+1)1}, Y_{(m_1+1)2})$, $(Y_{(m_1+2)1}, Y_{(m_1+2)2})$, ..., $(Y_{(m_1+m_2)1}, Y_{(m_1+m_2)2})$. So when $(Y_{01}, Y_{02}) \neq (0^n, 0^n)$, the probability that $Y_{m_1 2} = X_{(m_1+m_2)1} \oplus Y_{(m_1+m_2)2}$, i.e., T_{2n} outputs a bit 1, is $1/2^n$.

Thus, for CASE-2, we have

$$\begin{aligned}
&\Pr\{T_{2n}[r] = 1\} \\
&= \Pr\{T_{2n}[r] = 1 \mid (Y_{01}, Y_{02}) = (0^n, 0^n)\} \cdot \Pr\{(Y_{01}, Y_{02}) = (0^n, 0^n)\} \\
&\quad + \Pr\{T_{2n}[r] = 1 \mid (Y_{01}, Y_{02}) \neq (0^n, 0^n)\} \cdot \Pr\{(Y_{01}, Y_{02}) \neq (0^n, 0^n)\} \\
&= 1 \cdot 1/2^{2n} + 1/2^n \cdot (1 - 1/2^{2n}) \\
&< 1/2^{n-1},
\end{aligned}$$

where $r \in_R \mathcal{H}_{2n}$. This completes the proof. □

As a consequence of Theorem 1, we know that none of $\psi(f, f, f)$, $\psi(f, f, f^{(2)})$ and $\psi(f^{(2)}, f, f)$, where $f \in_R \mathcal{H}_n$, are robust.

Next we discuss the optimality of $\psi(g, g, f)$ and $\psi(g, f, f)$ where $f, g \in_R \mathcal{H}_n$. Apparently, $g_i(L, R) = (R \oplus f_i(L), L)$ can be distinguished from an $r \in_R \mathcal{H}_{2n}$. It was proved in [LR88] that *two* applications of Feistel-type transformations cannot obtain a robust permutation. In particular, Luby and Rackoff showed that $\psi(g, f)$, where $f, g \in \mathcal{H}_n$, can be easily distinguished from an $r \in_R \mathcal{H}_{2n}$.

Thus, by putting together Theorem 1 and Ohnishi's observations mentioned above, we see that to get a robust permutation in $\mathcal{H}_{2n}$, *two* independent random functions from $\mathcal{H}_n$ and three applications of Feistel-type transformations are not only *sufficient* but also *necessary*, as far as our construction is restricted to the permutations $\psi(f_3^{(k)}, f_2^{(j)}, f_1^{(i)})$, where $i, j, k \in \mathcal{N}$ and $f_1, f_2, f_3 \in \mathcal{H}_n$ such that for any $1 \leq s, t \leq 3$, either $f_s = f_t$ or f_s is independent of f_t. In other words, under the above condition, robust permutations $\psi(g, f, f)$ and $\psi(g, g, f)$ proposed by Ohnishi [Oh88], where $f, g \in_R \mathcal{H}_n$, are *optimal* in the sense that they consist of the minimal rounds of Feistel-type transformations, and "consume" the minimal number of independent random functions from $\mathcal{H}_n$.

2.2 Extensions

This section extends in two directions Theorem 1 to the case of generalized Feistel-type transformations.

Let $\ell \in \mathcal{N}$ with $\ell \geq 2$. Following [FNS75] (pp.1547-1549) and [Sch88], we associate with an $f_i \in \mathcal{H}_n$ a function $g_{1,i} \in \mathcal{H}_{\ell n}$ defined by

$$g_{1,i}(B_1, B_2, \ldots, B_\ell) = (B_2 \oplus f_i(B_1), B_3, \ldots, B_\ell, B_1),$$

where $B_i \in \Sigma^n$. See Figure 2.2. For simplicity, we call $g_{1,i}$ a *Type-1 transformation* (associated with f_i). Type-1 transformations will be discussed in more detail in Chapter 3.

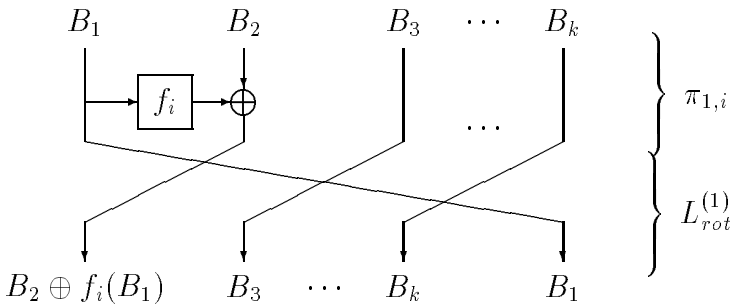


Figure 2.2: Type-1 Transformation

For $f_1, f_2, \ldots, f_s \in \mathcal{H}_n$, define $\theta(f_s, \ldots, f_2, f_1) = g_{1,s} \circ \cdots \circ g_{1,2} \circ g_{1,1}$. It is easy to show that when $s < 2\ell - 1$, $\theta(f_s, \ldots, f_2, f_1)$ can be distinguished from an $r \in_R \mathcal{H}_{\ell n}$. In Theorem 4, we will prove that when $s = 2\ell - 1$, $\theta(f_s, \ldots, f_2, f_1)$ is robust where $f_1, f_2, \ldots, f_s \in_R \mathcal{H}_n$.

Now we prove an impossibility result on $\theta(f_{2\ell-1}, \ldots, f_2, f_1)$. For $(2\ell - 1)$ integers $i_1, i_2, \ldots, i_{2\ell-1} \in \mathcal{N}$, let $\Theta(i_1, i_2, \ldots, i_{2\ell-1}) = \{\Theta_{\ell n}(i_1, i_2, \ldots, i_{2\ell-1}) \mid n \in \mathcal{N}\}$ be a PFG, where for each $n \in \mathcal{N}$, $\Theta_{\ell n}(i_1, i_2, \ldots, i_{2\ell-1})$ is a function that on input the description of $f \in \mathcal{H}_n$ outputs the description of $\theta(f^{(i_{2\ell-1})}, \ldots, f^{(i_2)}, f^{(i_1)}) \in \mathcal{H}_{\ell n}$.

Theorem 2 *For any $i_1, i_2, \ldots, i_{2\ell-1} \in \mathcal{N}$, there is a distinguisher $T = \{T_{\ell n} \mid n \in \mathcal{N}\}$ for $\Theta(i_1, i_2, \ldots, i_{2\ell-1})$. Each $T_{\ell n}$ has $(m_1 + m_2 + 1)$ oracle gates, where $m_1 = (i_1 + i_2 + \cdots + i_\ell)/d$, $m_2 = (i_2 + i_3 + \cdots + i_{\ell+1})/d$ and $d = \gcd(i_1 + i_2 + \cdots + i_\ell, i_2 + i_3 + \cdots + i_{\ell+1})$.*

Proof : There are two cases to be treated: $\ell = 2$ and $\ell > 2$. The former has been proved in Theorem 1. The proof for the latter is similar to that for the former.

As in the proof of Theorem 1, denote by $O_0, O_1, O_2, \ldots, O_{m_1+m_2}$ the (m_1+m_2+1) oracle gates, by $(X_{s1}, X_{s2}, \ldots, X_{s\ell})$ and $(Y_{s1}, Y_{s2}, \ldots, Y_{s\ell})$ the input to and output of O_s respectively, and by 0^n the all-0 string in Σ^n.

DESCRIPTION OF $T_{\ell n}$:

1. The input to O_0 is $(X_{01}, X_{02}, \ldots, X_{0\ell}) = (0^n, 0^n, \ldots, 0^n)$.
2. The input to O_1 is $(X_{11}, X_{12}, \ldots, X_{1\ell}) = (0^n, Y_{03}, 0^n, \ldots, 0^n)$. And if $m_1 > 1$, then for each $1 < p \leq m_1$, the input to O_p is $(X_{p1}, X_{p2}, \ldots, X_{p\ell}) = (0^n, X_{(p-1)2} \oplus Y_{(p-1)3}, 0^n, \ldots, 0^n)$.
3. The input to O_{m_1+1} is $(X_{(m_1+1)1}, X_{(m_1+1)2}, \ldots, X_{(m_1+1)\ell}) = (Y_{02}, 0^n, \ldots, 0^n)$. And if $m_2 > 1$, then for each $m_1 + 1 < t \leq m_1 + m_2$, the input to O_t is $(X_{t1}, X_{t2}, \ldots, X_{t\ell}) = (X_{(t-1)1} \oplus Y_{(t-1)2}, 0^n, \ldots, 0^n)$.
4. $T_{\ell n}$ outputs a bit 1 iff $Y_{m_1 2} = X_{(m_1+m_2)1} \oplus Y_{(m_1+m_2)2}$.

See also Figure 2.3 for the structure of $T_{\ell n}$. Analysis necessary is also similar to Theorem 1, and omitted here. □

Further analysis of the proof for Theorem 2 reveals that even given $(\ell - 1)$ independent random functions from $\mathcal{H}_n$, it is not guaranteed that one can always obtain robust permutations in $\mathcal{H}_{\ell n}$, by $(2\ell - 1)$ applications of generalized DES-like transformations. This is formally stated below.

For $(\ell + 1)$ integers $i_1, i_2, \ldots, i_{\ell+1} \in \mathcal{N}$, let

$$\overline{\Theta}(i_1, i_2, \ldots, i_{\ell+1}) = \{\overline{\Theta}_{\ell n}(i_1, i_2, \ldots, i_{\ell+1}) \mid n \in \mathcal{N}\}$$

be a PFG, where for each $n \in \mathcal{N}$, $\overline{\Theta}_{\ell n}(i_1, i_2, \ldots, i_{\ell+1})$ is a function that on input the descriptions of $f_1, f_2, \ldots, f_{\ell-1} \in \mathcal{H}_n$, outputs the description of

$$\theta(f_{\ell-1}, \ldots, f_3, f_2, f_1^{(i_{\ell+1})}, \ldots, f_1^{(i_2)}, f_1^{(i_1)}) \in \mathcal{H}_{\ell n}.$$

Theorem 3 *For any $i_1, i_2, \ldots, i_{\ell+1} \in \mathcal{N}$, $\overline{\Theta}(i_1, i_2, \ldots, i_{\ell+1})$ is not robust.*

2.3 Remarks

Our consideration has been restricted to the case of $\psi(f_3^{(k)}, f_2^{(j)}, f_1^{(i)})$ where $i, j, k \in \mathcal{N}$ and $f_1, f_2, f_3 \in \mathcal{H}_n$ such that for any $1 \leq s, t \leq 3$, either $f_s = f_t$ or f_s is independent of f_t. It is worth while examining other cases, such as $\psi(\hat{f}, f, f)$ and $\psi(f, f, \hat{f})$ where $\hat{f}$ is constructed from f with $\hat{f} \neq f^{(m)}$ for any $m \in \mathcal{N}$.

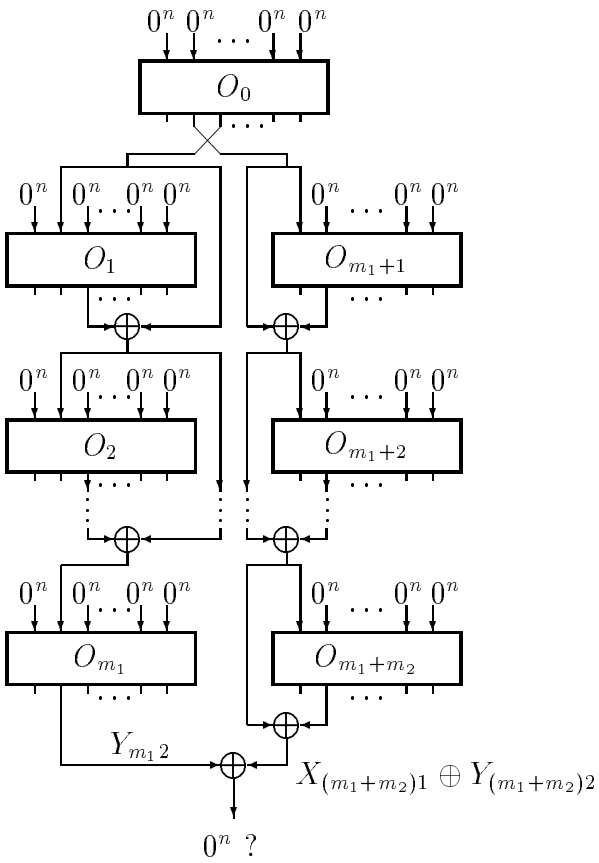


Figure 2.3: Structure of Oracle Circuit $T_{\ell n}$

In [ZMI89a], we posed the problem that *prove or disprove that* $\psi(f, f, f, f^{(2)})$ *and/or* $\psi(f^{(2)}, f, f, f)$ *are/is robust.* This problem has recently been answered affirmatively by Pieprzyk [Pie90].

Chapter 3

Cryptographically Useful Transformations

This chapter introduces various types of transformations, and generalizes FTT Lemma in many directions. First we introduce two operations on strings in Σ^{kn} — the *ρ-position left rotation* and the *ρ-position right rotation*. These two operations are denoted by $\mathrm{L}_{rot}^{(\rho)}$ and $\mathrm{R}_{rot}^{(\rho)}$, and defined as

$$\begin{aligned} \mathrm{L}_{rot}^{(\rho)}(B_1, B_2, \ldots, B_k) &= (B_{\rho+1}, \ldots, B_k, B_1, B_2, \ldots, B_\rho), \\ \mathrm{R}_{rot}^{(\rho)}(B_1, B_2, \cdots, B_k) &= (B_{k-\rho+1}, \ldots, B_k, B_1, B_2, \ldots, B_{k-\rho}) \end{aligned}$$

respectively, where $1 \leq \rho < k$ and $B_i \in \Sigma^n$. Note that both $\mathrm{L}_{rot}^{(\rho)}$ and $\mathrm{R}_{rot}^{(\rho)}$ are permutations on Σ^{kn}, and that $\mathrm{L}_{rot}^{(\rho)}$ is the inverse of $\mathrm{R}_{rot}^{(\rho)}$ and vice versa.

3.1 Various Transformations

3.1.1 Type-1 Transformations

This type of transformations has been defined in Chapter 2. For completeness, it is restated below. For each $f_i \in \mathcal{H}_n$, the Type-1 transformation associated with it is defined as

$$g_{1,i}(B_1, B_2, \ldots, B_k) = (B_2 \oplus f_i(B_1), B_3, \ldots, B_k, B_1),$$

where $B_i \in \Sigma^n$ for all $1 \leq i \leq k$. See Figure 2.2.

Note that $g_{1,i}$ can be decomposed into $g_{1,i} = \mathrm{L}_{rot}^{(1)} \circ \pi_{1,i}$ where $\pi_{1,i}$ is defined by

$$\pi_{1,i}(B_1, B_2, \ldots, B_k) = (B_1, B_2 \oplus f_i(B_1), B_3, \ldots, B_k).$$

It is easy to check that $\pi_{1,i} \circ \pi_{1,i}$ is the *identity transformation* on Σ^{kn}, i.e., $\pi_{1,i}$ is the inverse of itself. Such a function is usually called an *involution* [Kon81]. Now we see

that $g_{1,i}$ is an invertible permutation on Σ^{kn}, and its inverse, denoted by $\widetilde{g}_{1,i}$, is given by $\widetilde{g}_{1,i} = \pi_{1,i} \circ \mathrm{R}^{(1)}_{rot}$.

For $f_1, f_2, \ldots, f_s \in \mathcal{H}_n$, define $\psi_1(f_s, \ldots, f_2, f_1) = g_{1,s} \circ \cdots \circ g_{1,2} \circ g_{1,1}$. $\psi_1(f_s, \ldots, f_2, f_1)$ is also an invertible permutation on Σ^{kn}, and by definition, its inverse is

$$\widetilde{\psi}_1(f_s, \ldots, f_2, f_1) = \pi_{1,1} \circ \mathrm{R}^{(1)}_{rot} \circ \cdots \circ \pi_{1,s-1} \circ \mathrm{R}^{(1)}_{rot} \circ \pi_{1,s} \circ \mathrm{R}^{(1)}_{rot}.$$

3.1.2 Type-2 Transformations

Let $k = 2\ell$, where $\ell \in \mathcal{N}$. Associate with a function-tuple $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,2\ell-1})$, where $f_{i,j} \in \mathcal{H}_n$, a function $g_{2,i} \in \mathcal{H}_{kn}$ defined by

$$\begin{aligned} g_{2,i}(B_1, B_2, \ldots, B_k) &= (B_2 \oplus f_{i,1}(B_1), B_3, B_4 \oplus f_{i,3}(B_3), B_4, \ldots, \\ &\quad B_{k-1}, B_k \oplus f_{i,k-1}(B_{k-1}), B_1). \end{aligned}$$

$g_{2,i}$ is called a *Type-2 transformation*, and can be decomposed into $g_{2,i} = \mathrm{L}^{(1)}_{rot} \circ \pi_{2,i}$ where $\pi_{2,i}$ is defined by

$$\pi_{2,i}(B_1, B_2, \ldots, B_k) = (B_1, B_2 \oplus f_{i,1}(B_1), B_3, \ldots, B_{k-1}, B_k \oplus f_{i,k-1}(B_{k-1})).$$

See also Figure 3.1. Obviously, $\pi_{2,i}$ is an involution.

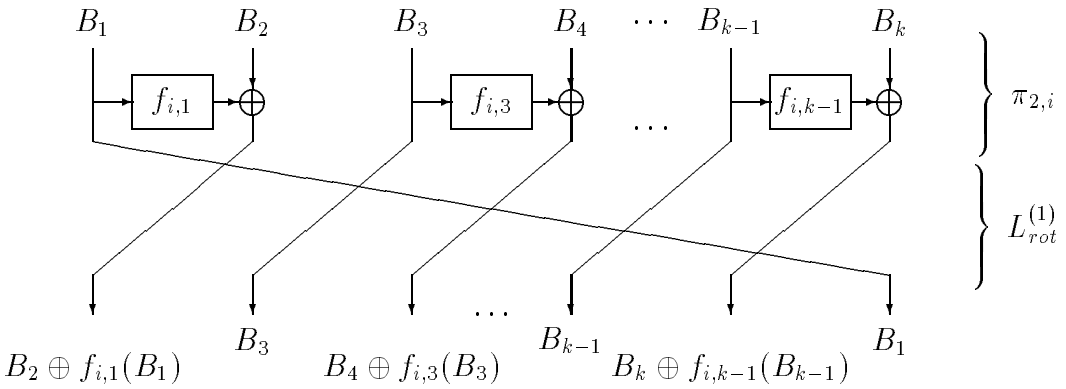


Figure 3.1: Type-2 Transformation

For s function-tuples $h_1, h_2, \ldots, h_s$, define $\psi_2(h_s, \ldots, h_2, h_1) = g_{2,s} \circ \cdots \circ g_{2,2} \circ g_{2,1}$. The inverse of $\psi_2(h_s, \ldots, h_2, h_1)$ is

$$\widetilde{\psi}_1(h_s, \ldots, h_2, h_1) = \widetilde{g}_{2,1} \circ \cdots \circ \widetilde{g}_{2,s-1} \circ \widetilde{g}_{2,s},$$

where $\widetilde{g}_{2,i} = \pi_{2,i} \circ \mathrm{R}^{(\rho)}_{rot}$.

3.1.3 Type-3 Transformations

Associate with a function-tuple $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$, where $f_{i,j} \in \mathcal{H}_n$, a function $g_{3,i} \in \mathcal{H}_{kn}$ defined by

$$g_{3,i}(B_1, B_2, \ldots, B_k) = (B_2 \oplus f_{i,1}(B_1), B_3 \oplus f_{i,2}(B_2), \ldots, B_k \oplus f_{i,k-1}(B_{k-1}), B_1).$$

Call $g_{3,i}$ a *Type-3 transformation*. We decompose $g_{3,i}$ into $g_{3,i} = \mathrm{L}_{rot}^{(1)} \circ \pi_{3,i}$ where $\pi_{3,i}$ is defined by

$$\pi_{3,i}(B_1, B_2, \ldots, B_k) = (B_1, B_2 \oplus f_{i,1}(B_1), B_3 \oplus f_{i,2}(B_2), \ldots, B_k \oplus f_{i,k-1}(B_{k-1})).$$

See Figure 3.2. $\pi_{3,i}$ is a permutation and its inverse is given by $\widetilde{\pi}_{3,i}(C_1, C_2, \cdots, C_k) = (B_1, B_2, \cdots, B_k)$, where $B_1 = C_1$ and $B_j = C_j \oplus f_{i,j-1}(B_{j-1})$ for each $2 \leq j \leq k$. One can soon find that $\pi_{3,i}$ is *not* an involution (Figure 3.3).

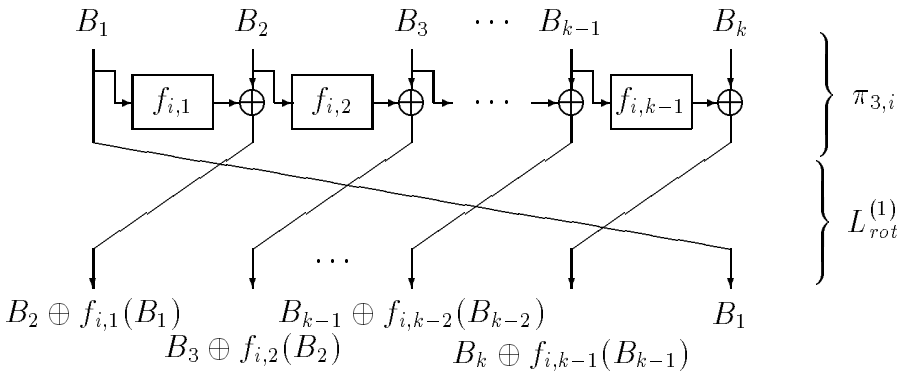


Figure 3.2: Type-3 Transformation

For s function-tuples $h_1, h_2, \ldots, h_s$, define $\psi_3(h_s, \ldots, h_2, h_1) = g_{3,s} \circ \cdots \circ g_{3,2} \circ g_{3,1}$. Since both $\pi_{3,i}$ and $\mathrm{L}_{rot}^{(1)}$ are permutations, hence so are $g_{3,i}$ and $\psi_3(h_s, \ldots, h_2, h_1)$. The inverse of $\psi_3(h_s, \ldots, h_2, h_1)$ is

$$\widetilde{\psi}_3(h_s, \ldots, h_2, h_1) = \widetilde{\pi}_{3,1} \circ \mathrm{R}_{rot}^{(1)} \circ \cdots \circ \widetilde{\pi}_{3,s-1} \circ \mathrm{R}_{rot}^{(1)} \circ \widetilde{\pi}_{3,s} \circ \mathrm{R}_{rot}^{(1)}.$$

3.1.4 Generalized Transformations

From its definition, we see that $\pi_{1,i}$ can be obtained from $\pi_{3,i}$ by dropping functions $f_{i,j}$ in $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$ for all $2 \leq j \leq k-1$. Similarly, when k is even, $\pi_{2,i}$ can also be obtained from $\pi_{3,i}$ by dropping functions $f_{i,j}$ in $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$ for all *even* $1 < j < k-1$.

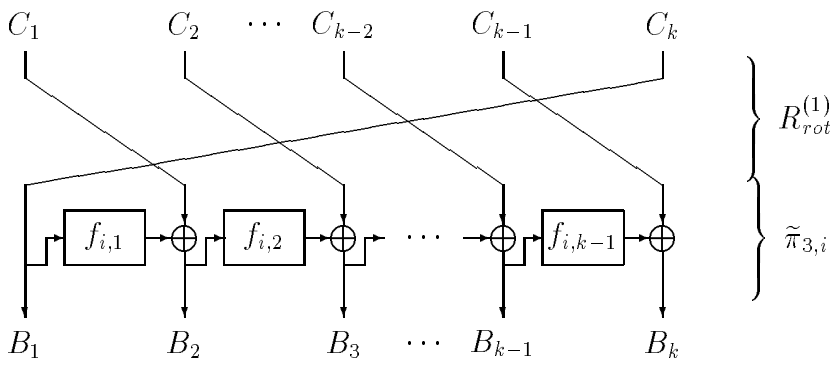


Figure 3.3: Inverse of Type-3 Transformation

Denote by $\pi_{\tau,i}$ a permutation obtained from $\pi_{3,i}$, by dropping certain functions $f_{i,j}$ in $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$. (Note: $\pi_{\tau,i} = \pi_{3,i}$ when dropping no function.) Define

$$g^{(\rho)}_{\tau,i} = \mathrm{L}^{(\rho)}_{rot} \circ \pi_{\tau,i},$$

where ρ is an integer with $1 \leq \rho \leq k-1$. Call transformations so obtained *Generalized Type-τ transformations*. Likewise, for s functions/function-tuples $h_1, h_2, \ldots, h_s$, define $\psi^{(\rho)}_{\tau}(h_s, \ldots, h_2, h_1) = g^{(\rho)}_{\tau,s} \circ \cdots \circ g^{(\rho)}_{\tau,2} \circ g^{(\rho)}_{\tau,1}$.

3.2 Theorems on the Transformations

Let E be a permutation consisting of $2k-1$ rounds of Type-1, or $k+1$ rounds of Type-2, or $k+1$ rounds of Type-3 transformations, each of which is chosen randomly and independently. The following Theorems 4 – 6 say that *no* oracle circuit with polynomially many oracle gates can distinguish between E and a truly random function.

Theorem 4 *Let Q be a polynomial in n and C_{kn} be an oracle circuit with $Q(n) < 2^n$ oracle gates. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_1(f_{2k-1}, \ldots, f_2, f_1)] = 1\}| \leq \frac{(k-1)Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{kn}$ and $f_1, f_2, \ldots, f_{2k-1} \in_R \mathcal{H}_n$.

Theorem 5 *Let Q be a polynomial in n and C_{kn} be an oracle circuit with $Q(n) < 2^n$ oracle gates where $k = 2\ell$. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2(h_{k+1}, \ldots, h_2, h_1)] = 1\}| \leq \frac{\ell^2 Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{kn}$ *and* $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ *with* $f_{i,j} \in_R \mathcal{H}_n$.

Theorem 6 *Let* Q *be a polynomial in* n *and* C_{kn} *be an oracle circuit with* $Q(n) < 2^n$ *oracle gates. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_3(h_{k+1}, \ldots, h_2, h_1)] = 1\}| \leq \frac{k(k-1)Q(n)^2}{2^{n+1}},$$

where $r \in_R \mathcal{H}_{kn}$ *and* $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$ *with* $f_{i,j} \in_R \mathcal{H}_n$.

Theorem 5 can be proved by essentially the same technique developed in [LR88] for proving FTT Lemma. For details see Section 3.3. Proofs for Theorems 4 and 6 can be derived from the proof for Theorem 5.

For Generalized Type-2 transformations, we have the following Theorem 7, which is crucial to our construction of block ciphers described in Chapter 4, and can be proved by modifying the proof for Theorem 5. For the other types of generalized transformations we have no results similar to Theorem 7. For reasons see Sections 3.4 – 3.5.

Theorem 7 *Let* $k = 2\ell$, *where* $\ell \in \mathcal{N}$, *and let* ρ *be an odd integer in* $[1, k]$. *Let* Q *be a polynomial in* n *and* C_{kn} *be an oracle circuit with* $Q(n) < 2^n$ *oracle gates. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2^{(\rho)}(h_{k+1}, \ldots, h_2, h_1)] = 1\}| \leq \frac{\ell^2 Q(n)^2}{2^n},$$

where $r \in_R H_{kn}$ *and* $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ *with* $f_{i,j} \in_R H_n$.

Theorems 4 – 7 are summarized in Table 3.1.

Table 3.1: Summary of Theorems 5 – 7

Transformations	Rounds	Probabilities
Type-1	$2k-1$	$\leq (k-1)Q(n)^2/2^n$
Type-2	$k+1$	$\leq \ell^2 Q(n)^2/2^n$
Type-3	$k+1$	$\leq k(k-1)Q(n)^2/2^{n+1}$
Generalized Type-2 (ρ: odd)	$k+1$	$\leq \ell^2 Q(n)^2/2^n$

3.3 Proof for Theorem 5

Assume that C_{kn} is an oracle circuit with $Q(n)$ oracle gates. Also assume that the Q oracle gates of C_{kn} are numbered by $1, 2, \ldots, Q(n)$. For simplicity, $Q(n)$ is abbreviated as Q. Also suppose that inputs to the oracle gates are all different; otherwise we can construct an oracle circuit C'_{kn} such that

1. C'_{kn} is functionally equivalent to C_{kn},
2. the number of oracle gates of C'_{kn} is Q and inputs to the oracle gates of C'_{kn} are mutually different, and
3. the size of C'_{kn} is at most as large as the cube of that of C_{kn} (See [LR88]).

The proof technique used here is essentially the same as that developed in [LR88] for proving FTT Lemma (called Main Lemma there).

Note that $k = 2\ell$ and that $\psi_2(h_{k+1}, \ldots, h_2, h_1)$ consists of $k+1$ Type-2 transformations, each of which has ℓ independent random functions from $\mathcal{H}_n$, where $\mathcal{H}_n$ denotes the set of all 2^{n2^n} functions from Σ^n to Σ^n.

Let Ω be the probability space with sample space $\Sigma^{Q(2\ell+1)\ell n}$ and the *uniform probability distribution*. For $\omega \in \Omega$, write it as $\omega = \omega_1\omega_2 \cdots \omega_{Q(2\ell+1)\ell n}$.

For each $1 \le i \le Q, 1 \le j \le 2\ell+1$ and odd t with $1 \le t \le 2\ell$, define a function $X_{i,j,t} : \Omega \to \Sigma^n$ as follows : $X_{i,j,t}(\omega) = \omega_{b+1} \cdots \omega_{b+n}$ where

$$b = [(j-1)\ell + \frac{t-1}{2}]Qn + (i-1)n.$$

There are totally $Q(2\ell+1)\ell$ such functions. For each $1 \le j \le 2\ell+1$ and odd t with $1 \le t \le 2\ell$, let $X_{j,t}(\omega) = X_{1,j,t}(\omega)X_{2,j,t}(\omega)\cdots X_{Q,j,t}(\omega)$.

Let $\omega \in \Omega$ be a sample point. For each $1 \le i \le Q$, replace the ith oracle gate by the following gate P_i. Note that the structure of P_i is similar to that of $\psi_2(h_{2\ell+1}, \ldots, h_2, h_1)$.

Gate P_i:

The input is $(B_{i,1}(\omega), B_{i,2}(\omega), \ldots, B_{i,2\ell}(\omega))$. Rename it as $(\alpha^i_{0,1}, \alpha^i_{0,2}, \ldots, \alpha^i_{0,2\ell})$.

1.1 **For each odd** $1 \le t \le 2\ell$, **compute** $u(i,1,t) = \min\{d | 1 \le d \le i, \alpha^d_{0,t} = \alpha^i_{0,t}\}$, and **let** $\alpha^i_{1,t} = \alpha^i_{0,t+1} \oplus X_{u(i,1,t),1,t}(\omega)$.

1.2 **Let** $\alpha^i_{1,2\ell} = \alpha^i_{0,1}$, and **for each even** $1 < t < 2\ell$, **let** $\alpha^i_{1,t} = \alpha^i_{0,t+1}$.

2.1 **For each odd** $1 \le t \le 2\ell$, **compute** $u(i,2,t) = \min\{d | 1 \le d \le i, \alpha^d_{1,t} = \alpha^i_{1,t}\}$, and **let** $\alpha^i_{2,t} = \alpha^i_{1,t+1} \oplus X_{u(i,2,t),2,t}(\omega)$.

2.2 **Let** $\alpha^i_{2,2\ell} = \alpha^i_{1,1}$, and **for each even** $1 < t < 2\ell$, **let** $\alpha^i_{2,t} = \alpha^i_{1,t+1}$.

$\vdots$

j.1 **For each odd** $1 \le t \le 2\ell$, **compute** $u(i,j,t) = \min\{d | 1 \le d \le i, \alpha^d_{j-1,t} = \alpha^i_{j-1,t}\}$, and **let** $\alpha^i_{j,t} = \alpha^i_{j-1,t+1} \oplus X_{u(i,j,t),j,t}(\omega)$.

j.2 **Let** $\alpha^i_{j,2\ell} = \alpha^i_{j-1,1}$, and **for each even** $1 < t < 2\ell$, **let** $\alpha^i_{j,t} = \alpha^i_{j-1,t+1}$.

$\vdots$

$(2\ell+1)$.1 **For each odd** $1 \le t \le 2\ell$, **compute** $u(i,2\ell+1,t) = \min\{d | 1 \le d \le i, \alpha^d_{2\ell,t} = \alpha^i_{2\ell,t}\}$, and **let** $\alpha^i_{2\ell+1,t} = \alpha^i_{2\ell,t+1} \oplus X_{u(i,2\ell+1,t),2\ell+1,t}(\omega)$.

$(2\ell+1)$.2 **Let** $\alpha^i_{2\ell+1,2\ell} = \alpha^i_{2\ell,1}$, and **for each even** $1 < t < 2\ell$, **let** $\alpha^i_{2\ell+1,t} = \alpha^i_{2\ell,t+1}$.

The output is $(\alpha^i_{2\ell+1,1}\alpha^i_{2\ell+1,2}\cdots\alpha^i_{2\ell+1,2\ell})$.

Let $B(\omega)$ be the output of the circuit C_{kn} when the oracle gates are computed as the above introduced gates P_i, and let $E(B)$ be the expectation of $B(\omega)$. Note that the value of $E(B)$ is equal to the probability that $B(\omega) = 1$. Thus, we have

$$E(B) = \Pr\{C_{kn}[\psi_2(h_{k+1}, \ldots, h_2, h_1)] = 1\}.$$

Now we introduce another kind of gates — P'_i. Each P'_i is obtained by replacing the "and **let** $\cdots$" parts of steps $2.1, \ldots, j.1, \ldots, (2\ell+1).1$ in the description of P_i by the following ones respectively:

$$\begin{aligned}
\text{and } \textbf{let } X'_{i,2,t}(\omega) &= \alpha^i_{1,t+1} \oplus X_{i,2,t}(\omega), \alpha^i_{2,t} = \alpha^i_{1,t+1} \oplus X'_{u(i,2,t),2,t}(\omega). \\
&\vdots \\
\text{and } \textbf{let } X'_{i,j,t}(\omega) &= \alpha^i_{j-1,t+1} \oplus X_{i,j,t}(\omega), \alpha^i_{j,t} = \alpha^i_{j-1,t+1} \oplus X_{u(i,j,t),j,t}(\omega). \\
&\vdots \\
\text{and } \textbf{let } X'_{i,2\ell+1,t}(\omega) &= \alpha^i_{2\ell,t+1} \oplus X_{i,2\ell+1,t}(\omega), \alpha^i_{2\ell+1,t} = \alpha^i_{2\ell,t+1} \oplus X'_{u(i,2\ell+1,t),2\ell+1,t}(\omega).
\end{aligned}$$

Let $B'(\omega)$ be the output bit of the circuit C_{kn} when the oracle gates are computed as the gates P'_i, and let $E(B')$ be the expectation of $B'(\omega)$.

Now we show that $E(B) = E(B')$. Note that $X_{i,j,t}(\omega)$ is identically and uniformly distributed, and that $\alpha^i_{j-1,t+1}$ does not depend on $X_{i,j,t}(\omega)$, Thus $X'_{i,j,t}(\omega) = \alpha^i_{j-1,t+1} \oplus X_{i,j,t}(\omega)$ is an identically and randomly chosen string from Σ^n. Consequently, $E(B)$ and $E(B')$ are identical.

Let R_i be a gate obtained in the following way: R_i is the same as P'_i except that the output is $(X_{i,2\ell+1,1}(\omega),\ X_{i,2\ell,3}(\omega),\ X_{i,2\ell+1,3}(\omega),\ X_{i,2\ell,5}(\omega),\ X_{i,2\ell+1,5}(\omega),\ \cdots,\ X_{i,2\ell+1,2\ell-1}(\omega),\ X_{i,2\ell,1}(\omega))$. Let $A(\omega)$ be the output bit of the circuit C_{kn} when the oracle gates are computed as the gates R_i, and let $E(A)$ be the expectation of $A(\omega)$. Like $E(B)$, the value of $E(A)$ is equal to the probability that $A(\omega) = 1$. Thus, $E(A) = \Pr\{C_{kn}[r] = 1\}$.

From above discussions, it follows that

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2(h_{k+1}, \ldots, h_2, h_1)] = 1\}| = |E(A) - E(B')|.$$

On the other hand,

$$|E(A) - E(B')| = \sum_{\omega \in \Omega} |A(\omega) - B'(\omega)|/2^{Q(2\ell+1)\ell n}.$$

Thus we need only to prove that

$$\sum_{\omega \in \Omega} |A(\omega) - B'(\omega)|/2^{Q(2\ell+1)\ell n} \leq \ell^2 Q^2/2^n.$$

Consider the case where the oracle gates of C_{kn} are computed as the gates R_i. For $\omega \in \Omega$ and for odd t with $1 \leq t \leq 2\ell$, call $X_{1,t}(\omega)$ *bad* if there are pairs (d, i) with $1 \leq d < i \leq Q$ such that $\alpha^d_{1,t} = \alpha^i_{1,t}$. By the same argument as for the claim in the bottom of [LR88] (p.383), we get

$$\Pr\{X_{1,t}(\omega) \text{ bad}\} \leq Q^2/2^{n+1}.$$

Similarly, for $\omega \in \Omega$, $2 \leq j \leq 2\ell$ and odd t with $1 \leq t \leq 2\ell$, call $X_{j,t}(\omega)$ *bad* if there are pairs (d, i) with $1 \leq d < i \leq Q$ such that $X_{d,j,t}(\omega) = X_{i,j,t}(\omega)$. Clearly,

$$\begin{aligned} \Pr\{X_{j,t}(\omega) \text{ bad}\} &\leq \sum_{2 \leq i \leq Q} (i-1)/2^n \\ &= Q(Q-1)/2^{n+1} < Q^2/2^{n+1}. \end{aligned}$$

Now we have a fact: *If $X_{j,t}(\omega)$ are not bad for all $1 \leq j \leq 2\ell$ and odd t with $1 \leq t \leq 2\ell$, then $A(\omega) = B'(\omega)$.* The reason is as follows. $X_{1,t}(\omega)$ is not bad which implies $u(i, 2, t) = i$ which implies $\alpha^i_{2,t} = X_{i,2,t}(\omega)$. $X_{2,t}(\omega)$ is not bad which implies

$u(i,3,t) = i$ which implies $\alpha^i_{3,t} = X_{i,3,t}(\omega)$. ..... $X_{2\ell,t}(\omega)$ is not bad which implies $u(i,2\ell+1,t) = i$ which implies $\alpha^i_{2\ell+1,t} = X_{i,2\ell+1,t}(\omega)$. These imply that the output of C_{kn} when the oracle gates are computed as the gates R_i and the output computed as P'_i are identical. In other words, $A(\omega) = B'(\omega)$.

Let Ω_g be the set of $\omega \in \Omega$ such that $X_{j,t}(\omega)$ are not bad for all $1 \le j \le 2\ell$ and odd t with $1 \le t \le 2\ell$, and let $\Omega_b = \Omega - \Omega_g$. Recall that $\sharp S$ denotes the number of elements in a set S. Thus we have

$$\begin{aligned}
\sharp\Omega_b/\sharp\Omega &= \Pr\{X_{j,t}(\omega) \text{ bad for some } 1 \le j \le 2\ell \text{ and odd } t \text{ with } 1 \le t \le 2\ell\} \\
&= \sum_{\substack{1\le j\le 2\ell \\ t=1,3,\ldots,2\ell-1}} \Pr\{X_{j,t}(\omega) \text{ bad}\} \\
&\le \ell^2 Q^2/2^n,
\end{aligned}$$

and

$$\begin{aligned}
&|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2(h_{k+1},\ldots,h_2,h_1)] = 1\}| \\
&= \sum_{\omega\in\Omega} |A(\omega) - B'(\omega)|/2^{Q(2\ell+1)\ell n} \\
&= \Big(\sum_{\omega\in\Omega_g} |A(\omega) - B'(\omega)| + \sum_{\omega\in\Omega_b} |A(\omega) - B'(\omega)|\Big)/2^{Q(2\ell+1)\ell n} \\
&= \sum_{\omega\in\Omega_b} |A(\omega) - B'(\omega)|/2^{Q(2\ell+1)\ell n} \\
&\le \sum_{\omega\in\Omega_b} 1/2^{Q(2\ell+1)\ell n} \\
&= \sharp\Omega_b/\sharp\Omega \\
&\le \ell^2 Q^2/2^n.
\end{aligned}$$

This completes the proof of Theorem 5.

3.4 Minimum Rounds for Robustness

This section discusses minimum rounds for achieving robustness when a permutation is constructed from some kind of transformations.

First we introduce a useful lemma.

Lemma 2 *Let P be a subset of $\mathcal{H}_{kn}$. If for any function $p \in P$ and for any input $s = (s_i, s_\ell, s_j) \in \Sigma^{kn}$, the output of p takes the form of*

$$(\ldots, s_\ell \oplus (\cdots \widehat{s_\ell} \cdots), \ldots),$$

where $0 \leq i, \ell, j \leq kn, \ell \geq 1, i + \ell + j = kn$, $s_i \in \Sigma^i, s_\ell \in \Sigma^\ell, s_j \in \Sigma^j$ *and* $(\cdots \widehat{s_\ell} \cdots)$ *means that the string does not depend on* s_ℓ*, then there is a simple oracle circuit distinguishing between a function* $p \in P$ *and a function randomly and uniformly selected from* $\mathcal{H}_{kn}$.

Proof : Our oracle circuit has two oracle gates. It works as follows: Select two strings $s_1 = (s_i, s'_\ell, s_j)$ and $s_2 = (s_i, s''_\ell, s_j)$ from Σ^{kn} such that $s'_\ell \neq s''_\ell$. Then input s_1 to the first oracle gate and s_2 to the second oracle gate. Assume the outputs of the two oracle gates are c_1 and c_2 respectively. Our oracle circuit outputs a bit 1 iff the XOR of c_1 and c_2 is $(\ldots, s'_\ell \oplus s''_\ell, \ldots)$.

When the oracle gates are evaluated by a function $p \in P$, we have $c_1 = (\ldots, s'_\ell \oplus (\cdots \widehat{s'_\ell} \cdots), \ldots)$ and $c_2 = (\ldots, s''_\ell \oplus (\cdots \widehat{s''_\ell} \cdots), \ldots)$, respectively. And the XOR of c_1 and c_2 is always $(\ldots, s'_\ell \oplus s''_\ell, \ldots)$. Thus with probability 1 the oracle circuit outputs 1.

On the other hand, when the oracle gates are evaluated by a function randomly and uniformly selected from $\mathcal{H}_{kn}$, the probability that the XOR of c_1 and c_2 is equal to $(\ldots, s'_\ell \oplus s''_\ell, \ldots)$ is $\frac{1}{2^\ell}$. □

Recall that for a function-tuple $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$, where $f_{i,j} \in \mathcal{H}_n$, a Type-3 transformation is defined as $g_{3,i} = \mathrm{L}^{(1)}_{rot} \circ \pi_{3,i}$, where

$$\pi_{3,i}(B_1, B_2, \ldots, B_k) = (B_1, B_2 \oplus f_{i,1}(B_1), B_3 \oplus f_{i,2}(B_2), \ldots, B_k \oplus f_{i,k-1}(B_{k-1})),$$

and that for s function-tuples $h_1, h_2, \ldots, h_s$, a permutation consisting of s rounds of Type-3 transformations is defined as $\psi_3(h_s, \ldots, h_2, h_1) = g_{3,s} \circ \cdots \circ g_{3,2} \circ g_{3,1}$.

Let $\pi_{\tau,i}$ be a permutation obtained from $\pi_{3,i}$ by dropping certain functions $f_{i,j}$ in $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$. ($\pi_{\tau,i} = \pi_{3,i}$ when dropping no function.) Define $g_{\tau,i} = \mathrm{L}^{(1)}_{rot} \circ \pi_{\tau,i}$, and call it a Type-$\tau$ transformation. Also let $\psi_\tau(h_s, \ldots, h_2, h_1)$ be a permutation in $\mathcal{H}_{kn}$ consisting of s rounds of Type-τ transformations.

Let λ_τ denote the minimum number s at which $\psi_\tau(h_s, \ldots, h_2, h_1)$ is robust. Now we are in a position to prove that $\lambda_\tau \geq k+1$, i.e., a necessary condition for $\psi_\tau(h_s, \ldots, h_1, h_1)$ to be robust is that $s \geq k + 1$.

Theorem 8 *The following inequalities hold:*

(1) $\lambda_\tau \geq k + 1$.

(2) $\lambda_1 \geq 2k - 1$.

Proof : (1) Consider Type-3 transformations. Let $(B_1, B_2, \ldots, B_k)$, where $B_i \in \Sigma^n$, be an input string to $\psi_3(h_s, \ldots, h_2, h_1)$. Call B_1 the first substring of $(B_1, B_2, \ldots, B_k)$, B_2 the second substring of $(B_1, B_2, \ldots, B_k)$ and so on. Focus our attention on the last substring B_k of $(B_1, B_2, \ldots, B_k)$.

The output of $g_{3,1}$ is $(B_2 \oplus f_{1,1}(B_1), B_3 \oplus f_{1,2}(B_2), \ldots, B_k \oplus f_{1,k-1}(B_{k-1}), B_1)$. Note that B_k is XORed with $f_{1,k-1}(B_{k-1})$, where B_{k-1} is B_k's left neighborhood, and then the result is shifted to the $(k-1)$th position. Also note that *no* substring in the position 1 to the position $(k-2)$ of the output of $g_{3,1}$ depends on B_k. That is, substrings in the position 1 to the position $(k-2)$ of the output of $g_{3,1}$ take the forms of $(\cdots \widehat{B_k} \cdots)$.

Now B_k appears in the $(k-2)$th substring of the output of $g_{3,2}$, in the form of $B_k \oplus (\cdots \widehat{B_k} \cdots)$, and substrings in the position 1 to the position $(k-3)$ of the output of $g_{3,2}$ take the forms of $(\cdots \widehat{B_k} \cdots)$. So on, the first substring of the output of $g_{3,k-1}$, and hence the last substring of the output of $g_{3,k}$, is of the form $B_k \oplus (\cdots \widehat{B_k} \cdots)$.

Thus for any $1 \leq s \leq k$, there is a substring in the output of $\psi_3(h_s, \ldots, h_2, h_1)$ taking the form of $B_k \oplus (\cdots \widehat{B_k} \cdots)$. By Lemma 2, such a function in $\mathcal{H}_{kn}$ can not be robust. In other words, $\lambda_3 \geq k+1$.

Now consider a Type-τ transformation $g_{\tau,i} = \mathrm{L}^{(1)}_{rot} \circ \pi_{\tau,i}$. Since $\pi_{\tau,i}$ is obtained from $\pi_{3,i}$ by *dropping* certain functions $f_{i,j}$ in $h_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k-1})$, the total number of substrings in the output of $\psi_\tau(h_k, \ldots, h_2, h_1)$ that take the forms of $B_i \oplus (\cdots \widehat{B_i} \cdots)$ is not less than that of $\psi_3(h_k, \ldots, h_2, h_1)$. Consequently, $\lambda_\tau \geq \lambda_3 \geq k+1$.

(2) The last substring of the output of $g_{1,k}$ is of the form of $B_k \oplus (\cdots \widehat{B_k} \cdots)$. From round to round, this substring is left-shifted, one position by one position and with no change in form. Finally, it reaches the second position of the output of $g_{1,k+(k-2)} = g_{1,2k-2}$. By Lemma 2, $\lambda_1 \geq 2k-1$. □

A type of transformations is called *singular* if a transformation g of that type is defined as

$$g(B_1, \ldots, B_i, \ldots, B_k) = (C_1, \ldots, C_{j-1}, C_j, C_{j+1}, \ldots, C_k),$$

where $C_j = B_i$ and neither $(C_1, \ldots, C_{j-1})$ nor $(C_{j+1}, \ldots, C_k)$ depends on B_i. For example, Type-1 transformations are singular, but Type-2 and Type-3 ones are non-singular.

Theorem 9 *Let Q be a polynomial in n and C_{kn} be an oracle circuit with $Q(n) < 2^n$ oracle gates. Let $\psi_\tau(h_s, \ldots, h_2, h_1)$ be a permutation consisting of s rounds of* singular *Type-τ transformations, where $h_s, \ldots, h_2, h_1$ are independent random function tuples. Then*

(1) *when* $s \leq k+1$, $\psi_\tau(h_s, \ldots, h_2, h_1)$ *is not robust, and*

(2) *when* $s = 2k-1$,

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_\tau(h_s, \ldots, h_2, h_1)] = 1\}| \leq \frac{(k-1)^2 Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{kn}$.

Proof : The proof for (1) is similar to that for (1) of Theorem 8, and the proof for (2) to that for Theorem 5. □

Theorem 10 *Let* Q *be a polynomial in* n *and* C_{kn} *be an oracle circuit with* $Q(n) < 2^n$ *oracle gates, and let* $\psi_\tau(h_s, \ldots, h_2, h_1)$ *be a permutation consisting of* s *rounds of* non-singular *Type-*τ *transformations, where* $h_s, \ldots, h_2, h_1$ *are independent random function tuples. Then when* $s = k+1$,

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_\tau(h_s, \ldots, h_2, h_1)] = 1\}| \leq \frac{k(k-1)Q(n)^2}{2^{n+1}},$$

where $r \in_R \mathcal{H}_{kn}$.

Proof : Similar to that for Theorem 5. □

Now we consider Generalized Type-τ transformations

$$g_{\tau,i}^{(\rho)} = \mathrm{L}_{rot}^{(\rho)} \circ \pi_{\tau,i}.$$

Let $\psi_\tau^{(\rho)}(h_s, \ldots, h_2, h_1)$ be a permutation in $\mathcal{H}_{kn}$ consisting of s rounds of Generalized Type-τ transformations. Denote by $\lambda_\tau^{(\rho)}$ the minimum number s at which $\psi_\tau^{(\rho)}(h_s, \ldots, h_2, h_1)$ is robust, where h_i are independent random function tuples. In particular, $\lambda_\tau^{(\rho)}$ is defined to be $+\infty$ if $\psi_\tau^{(\rho)}(h_s, \ldots, h_2, h_1)$ is not robust no matter how large s is. We have the following theorem which is easy to prove.

Theorem 11 *The following statements holds:*

(1) $\lambda_\tau^{(\rho)} \geq \lambda_\tau \geq k+1$.

(2) $\lambda_2^{(\rho)} = \lambda_2$ *when* ρ *is an* odd *integer in* $[1, k]$, *and* $\lambda_2^{(\rho)} = +\infty$ *where* ρ *is an* even *integer in* $[1, k]$.

Main results of Section 3.4 are summarized in Table 3.2.

Table 3.2: Minimum Rounds λ for Robustness

Transformations	λ
Type-1	$2k-1$
Type-2 (k: even)	$k+1$
Type-3	$k+1$
Generalized Type-τ	$\geq k+1$

3.5 Optimal Transformations

In this section we examine what kinds of transformations are *optimal* in a reasonable sense to be defined below.

Informally, a type of transformations is *optimal* if

1. a permutation E consisting of $k+1$ rounds of randomly chosen transformations is indistinguishable from a truly random function by all oracle circuits with polynomially many oracle gates, and

2. the inverse of E can be computed in the same parallel time as E.

The computing procedures for Generalized Type-τ transformations $g_{\tau,i}^{(\rho)}$, and hence for $\psi_{\tau}^{(\rho)}(h_s,\ldots,h_2,h_1)$, can be represented by acyclic *computation graphs*. There are three kinds of nodes in a computation graph: input nodes, output nodes and internal nodes. Each internal node in a computation graph represents a generic operation : computing a function $f_{i,j}$ or XORing two strings.

The length of a path between two nodes is defined as the number of arcs in the path. Now assume that the length of the longest path(s) from input nodes to output nodes in a computation graph is L. Then

1. the *depth* of the graph is defined to be $L-1$.

2. the *normal-delay* $D^{+}(\rho,\tau,s)$ of a permutation $\psi_{\tau}^{(\rho)}(h_s,\ldots,h_2,h_1) \in \mathcal{H}_{kn}$ is defined as the depth of the computation graph for the permutation.

3. the *inverse-delay* $D^{-}(\rho,\tau,s)$ is defined as that for the inverse of the permutation.

4. the *sum-delay* $D(\rho,\tau,s)$ is defined as $D(\rho,\tau,s) = D^{+}(\rho,\tau,s) + D^{-}(\rho,\tau,s)$.

Clearly, $D^{+}(\rho,\tau,s) = 2s$, and $D^{-}(\rho,\tau,s) \geq D^{+}(\rho,\tau,s) \geq 2s$. Thus

$$\begin{aligned} D(\rho,\tau,s) &= D^{+}(\rho,\tau,s) + D^{-}(\rho,\tau,s) \\ &\geq 2D^{+}(\rho,\tau,s) \geq 4s. \end{aligned}$$

Recall that $\lambda_\tau^{(\rho)}$ denotes the minimum number of rounds s at which $\psi_\tau^{(\rho)}(h_s, \ldots, h_2, h_1)$ is robust. From Theorem 11, we have $\lambda_\tau^{(\rho)} \geq k+1$. Hence, $D(\rho, \tau, \lambda_\tau^{(\rho)}) \geq 4(k+1)$.

Definition 3 *A Generalized Type-τ transformation is called optimal if* $D(\rho, \tau, \lambda_\tau^{(\rho)}) = 4(k+1)$.

Now we discuss the optimality of transformations. First we have two facts:

1. When $g_{\tau,i}^{(\rho)} = \mathrm{L}_{rot}^{(\rho)} \circ \pi_{\tau,i}^{(\rho)}$ with $\pi_{\tau,i}^{(\rho)}$ being not an involution, we have $D^-(\rho, \tau, s) > 2s$. So, transformations like Type-3 cannot be optimal.

2. When $\pi_{\tau,i}^{(\rho)}$ is an involution but $g_{\tau,i}^{(\rho)}$ is singular, we have $\lambda_\tau^{(\rho)} > k+1$, and hence $D(\rho, \tau, \lambda_\tau^{(\rho)}) > 4(k+1)$. So transformations like Type-1 cannot be optimal.

Consider the following two cases: odd k and even k.

1. In the former case, either $g_{\tau,i}^{(\rho)} = \mathrm{L}_{rot}^{(\rho)} \circ \pi_{\tau,i}^{(\rho)}$ is singular or $\pi_{\tau,i}^{(\rho)}$ is not an involution. Thus by the above two facts, no optimal transformation can be obtained.

2. In the latter case, it is not hard to verify that the only non-singular transformations $g_{\tau,i}^{(\rho)} = \mathrm{L}_{rot}^{(\rho)} \circ \pi_{\tau,i}^{(\rho)}$ with $\pi_{\tau,i}^{(\rho)}$ being involutions are Generalized Type-2 ones with ρ odd. For such transformations we have $D(\rho, 2, \lambda_2^{(\rho)}) = 4(k+1)$.

Thus we have proved:

Theorem 12 *Among all types of transformations discussed in this paper, Generalized Type-2 transformations $g_{2,i}^{(\rho)} = \mathrm{L}_{rot}^{(\rho)} \circ \pi_{2,i}^{(\rho)}$ with even k and odd ρ, are the only optimal ones.*

3.6 Notion of Super-Robustness

Luby and Rackoff introduced also the notion of *super-robust pseudo-random permutation generators* in [LR88]. (They owed the notion to O. Goldreich.) Intuitively, a pseudo-random permutation generator is super-robust if no super-oracle circuit can tell a permutation randomly specified by the generator from a randomly and uniformly chosen one. A *super-oracle circuit* is an oracle circuit with two kinds of oracle gates. The first is called the *normal* oracle gates which are evaluated using some permutation, and the second the *inverse* oracle gates which are evaluated using the inverse of the permutation.

When a robust pseudo-random permutation generator is used to construct a block cipher, the cipher is secure against the chosen plaintext attack, but not necessarily

secure against the chosen plaintext/ciphertext attack. When a super-robust pseudo-random permutation generator is used to construct a block cipher, the cipher is secure against the chosen plaintext/ciphertext attack [LR88].

Luby and Rackoff showed that functions consisting of four rounds of FTT's are super-robust. We generalize the result to the following one.

Theorem 13 *Let $k = 2\ell$, where $\ell \in \mathcal{N}$, and let ρ be an odd integer in $[1, k]$. Assume that $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ consists of s rounds of Generalized Type-2 transformations, where $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ with $f_{i,j} \in_R \mathcal{H}_n$. Then $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ is super-robust iff $s \geq k + 2$.*

Proof: (1) When $s < k+1$, $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ is not robust, hence not super-robust. Next we prove that when $s = k + 1$, $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ is not super-robust. We do it by showing a super-oracle circuit C_{kn} which distinguishes $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ from a permutation randomly and uniformly chosen from $\mathcal{S}_{kn}$, the set of all permutations on Σ^{kn}.

Suppose that

$$C = (C_1, C_2, \ldots, C_k) = \psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)(B),$$

where $B = (B_1, B_2, \ldots, B_k)$ and $C_i, B_i \in \Sigma^n$. Note that for each $1 \leq i \leq k$, C_i is a function of $B_1, B_2, \ldots, B_k$. For each $1 \leq i, j \leq k$, denote by $MIN_B(i, j)$ the maximum number of functions applied to B_j in C_i. For example, if $C_2 = B_1 \oplus f_2(B_2 \oplus f_1(B_1))$ then $MIN_B(2, 1) = 2$ and $MIN_B(2, 2) = 1$.

Since $s = k + 1$, there must exist i_1, j such that $MIN_B(i_1, j) = 1$. Suppose that

$$C_{i_1} = B_e \oplus f_{xy}(B_j \oplus (\cdots \widehat{B_j} \cdots)) \oplus (\cdots),$$

where $(\cdots \widehat{B_j} \cdots)$ means that the string does not depend on B_j. There must also exist an i_2 such that $C_{i_2} = B_j \oplus (\cdots)$.

Our super-oracle circuit C_{kn} has two normal oracle gates and one inverse oracle gate. C_{kn} works as follows: Choose two strings $B = (B_1, B_2, \ldots, B_j, \ldots, B_k)$ and $B' = (B'_1, B'_2, \ldots, B'_j, \ldots, B'_k)$ from Σ^{kn} such that $B'_i = B_i$ for all $i \neq j$ and $B'_j \neq B_j$. Input B to the first normal oracle gate and B' to the second normal oracle gate. Assume the two corresponding outputs are $C = (C_1, C_2, \ldots, C_{i_2}, \ldots, C_k)$ and $C' = (C'_1, C'_2, \ldots, C'_{i_2}, \ldots, C'_k)$ respectively. Now let $C'' = (C''_1, C''_2, \ldots, C''_{i_2}, \ldots, C''_k)$ where $C''_{i_2} = C_{i_2} \oplus B_j \oplus B'_j$ and $C''_i = C_i$ for all $i \neq i_2$. Then input C'' to the inverse oracle gate. Assume that the output of the inverse oracle gate is $B'' = (B''_1, B''_2, \ldots, B''_e, \ldots, B''_k)$.

The super-oracle circuit C_{kn} outputs a single bit 1 iff $B''_e = B_e \oplus C_{i_1} \oplus C'_{i_1}$. Whenever $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ and its inverse are used to evaluate the oracle gates,

C_{kn} outputs 1 with probability 1; and whenever a random permutation and its inverse over Σ^{kn} are used to evaluate the oracle gates, C_{kn} outputs 1 with probability $\frac{1}{2^n}$.

(2) When $s \geq k+2$, super-robustness for $\psi_2^{(\rho)}(h_s, \ldots, h_2, h_1)$ can be proved in a similar way as for proving Theorem 2. $\square$

Chapter 4

Block Ciphers Based on Type-2 Transformations

Applying the only optimal Generalized Type-2 Transformations we construct a provably secure block cipher (PSBC) in this chapter. We also consider a variant of PSBC which uses robust PSG for key-expanding. Finally, we present four concrete block ciphers based on the the variant of PSBC.

4.1 A Few Observations

As pointed out in [Sch88], FTT Lemma remains true even if the number of oracle gates is replaced by $Q(n) \leq 2^{o(n)}$ [Sch88]. Here $Q(n) \leq 2^{o(n)}$ means that $Q(n) \leq 2^{f(n)}$ for some $f(n)$, which satisfies

$$\lim_{n\to\infty} \frac{cf(n)}{n} = 0$$

for every positive constant c, i.e., $f(n) = o(n)$.

Lemma 3 (FTT Lemma, Version 2, [Sch88]) *Let C_{2n} be an oracle circuit with $Q(n) \leq 2^{o(n)}$ oracle gates. Then*

$$|\Pr\{C_{2n}[r] = 1\} - \Pr\{C_{2n}[\psi(f_3, f_2, f_1)] = 1\}| \leq \frac{Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{2n}$ and $f_1, f_2, f_3 \in_R \mathcal{H}_n$.

Schnorr's observation also applies to our Theorem 7 stated in Section 3.2.

Theorem 14 *Let $k = 2\ell$, where $\ell \in \mathcal{N}$, and let ρ be an odd integer in $[1, k]$. Let C_{kn} be an oracle circuit with $Q(n) \leq 2^{o(n)}$ oracle gates. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2^{(\rho)}(h_{k+1}, \ldots, h_2, h_1)] = 1\}| \leq \frac{\ell^2 Q(n)^2}{2^n},$$

where $r \in_R \mathcal{H}_{kn}$ *and* $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ *with* $f_{i,j} \in_R \mathcal{H}_n$.

Next we make a few more observations. Let $t \in \mathcal{N}$ and $n = \lceil (\log t)^{1+\varepsilon} \rceil$ for some $\varepsilon \in (0, 1]$, where the logarithm is taken to the base 2. Then for any constants c and ε' with $c > 0$ and $0 \leq \varepsilon' < \varepsilon$, we have

$$c \log t \leq c(\log t)^{1+\varepsilon'} = o(\lceil (\log t)^{1+\varepsilon} \rceil) = o(n),$$

and

$$t^c = 2^{c \log t} \leq 2^{c(\log t)^{1+\varepsilon'}} = 2^{o(n)}.$$

Thus we obtain from Theorem 14 the following one.

Theorem 15 *Let* $k = 2\ell$, *where* $\ell \in \mathcal{N}$, *and let* ρ *be an odd integer in* $[1, k]$. *Assume that* $t \in \mathcal{N}$, $\varepsilon \in (0, 1]$, $n = \lceil (\log t)^{1+\varepsilon} \rceil$ *and* $Q(t) \leq 2^{o(n)}$ *is a polynomial in* t. *(Notice that* $2^n = 2^{\lceil (\log t)^{1+\varepsilon} \rceil}$ *is quasi-polynomial in* t.*)* [1] *Let* C_{2n} *be an oracle circuit with* $Q(t)$ *oracle gates. Then*

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2^{(\rho)}(h_{k+1}, \ldots, h_2, h_1)] = 1\}| \leq \frac{\ell^2 Q(t)^2}{2^n},$$

where $r \in_R \mathcal{H}_{kn}$ *and* $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ *with* $f_{i,j} \in_R \mathcal{H}_n$.

4.2 PSBC — A Provably Secure Block Cipher

Theorem 15 says that theoretically, if one is quasi-polynomially powerful, then one can construct a block cipher secure against any polynomially powerful adversary.

Let $n = \lceil (\log t)^{1+\varepsilon} \rceil$, where $t \in \mathcal{N}$ and $\varepsilon \in (0, 1]$. Let $\ell \in \mathcal{N}$, $k = 2\ell$ and ρ be an odd integer in $[1, k]$. Assume that the plaintext and ciphertext spaces are Σ^{kn}. Denote by $B = (B_1, B_2, \ldots, B_k)$ a plaintext in Σ^{kn} and by $C = (C_1, C_2, \ldots, C_k)$ the ciphertext of B, where $B_i, C_i \in \Sigma^n$.

PSBC consists principally of s rounds of Generalized Type-2 transformations where $s \geq k+2$. The reason for choosing $s \geq k+2$ is as follows: When $s = k+1$, our block cipher PSBC is secure against chosen plaintext attack, but not secure against chosen plaintext/ciphertext attack. When $s \geq k+2$, PSBC is secure against chosen plaintext/ciphertext attack. See Section 3.6 and [LR88].

The enciphering algorithm E for PSBC can be concisely expressed as

$$E = \pi_{2,s} \circ \mathrm{L}_{rot}^{(\rho)} \circ \cdots \circ \mathrm{L}_{rot}^{(\rho)} \circ \pi_{2,2} \circ \mathrm{L}_{rot}^{(\rho)} \circ \pi_{2,1}.$$

See Figure 4.1. The inverse of E is $\pi_{2,1} \circ \mathrm{R}_{rot}^{(\rho)} \circ \cdots \circ \mathrm{R}_{rot}^{(\rho)} \circ \pi_{2,s-1} \circ \mathrm{R}_{rot}^{(\rho)} \circ \pi_{2,s}$. So the deciphering algorithm D for PSBC is obtained from E by

[1] We call a function f *quasi-polynomial* in t if for any polynomial P, for any constant $c > 0$ and for all sufficiently large t, we have $P(t) < f(t) < 2^{t^c}$.

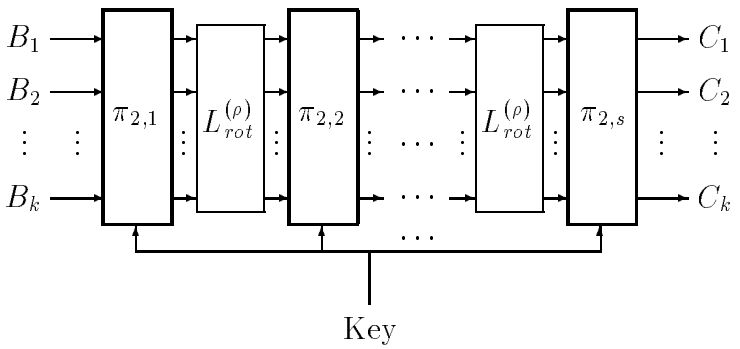


Figure 4.1: Enciphering Algorithm for PSBC

1. interchanging h_i with h_{s+1-i} for each $1 \leq i \leq \lfloor \frac{s}{2} \rfloor$, and
2. changing the mapping (or wiring) representing $\mathrm{L}_{rot}^{(\rho)}$ to the mapping (or wiring) representing $\mathrm{R}_{rot}^{(\rho)}$.

Notice that when ℓ is odd and $\rho = \ell$, there is no need for changing the mapping (or wiring), since $\mathrm{L}_{rot}^{(\rho)} = \mathrm{R}_{rot}^{(\rho)}$ in this case.

4.3 A Variant of PSBC

The block cipher PSBC described in Section 4.2 requires quasi-polynomially many memory cells for both enciphering and deciphering procedures. Thus it is practically impossible to realize the cipher.

In the remaining portion of this chapter we present a variant of PSBC, in order to pave the way to practically realizable ciphers. The variant is obtained by adding to PSBC a *key-expanding part*. The key-expanding part stretches a short string into a long one, i.e., is a PSG. The PSG we use is a *robust* one (see Definition 1), and it is essentially due to Ohnishi and Schnorr [Oh88, Sch88]. PSG's suggested by Maurer and Massey [MM89] may serve for our key-expanding purpose too. Using the recent result of Pieprzyk [Pie90], the length of keys in the variant of PSBC can be halved.

4.3.1 A Robust Pseudo-random String Generator

Ohnishi observed that FTT Lemma remains valid even when *two* independent random functions are available [Oh88].

Lemma 4 (FTT Lemma, Version 3, [Oh88]) *Let Q be a polynomial in n, and let C_{2n} be an oracle circuit with $Q(n) \leq 2^{o(n)}$ oracle gates. Then*

$$|\Pr\{C_{2n}[r] = 1\} - \Pr\{C_{2n}[\psi(f_2, f_1, f_1)] = 1\}| \leq \frac{2(Q(n)+1)^2}{2^n},$$

and

$$|\Pr\{C_{2n}[r] = 1\} - \Pr\{C_{2n}[\psi(f_2, f_2, f_1)] = 1\}| \leq \frac{2(Q(n)+1)^2}{2^n},$$

where $r \in_R \mathcal{H}_{2n}$, $f_1, f_2 \in_R \mathcal{H}_n$.

Schnorr [Sch88] showed that FTT Lemma implies that we can explicitly construct a PSG without any hypotheses. Putting together observations made in [Oh88] and in [Sch88], we have the following PSG.

First we note that there is a natural one-one correspondence between functions in $\mathcal{H}_n$ and strings in Σ^{n2^n}, i.e., a bijection Φ_n from $\mathcal{H}_n$ to Σ^{n2^n}. The bijection maps a function $f \in \mathcal{H}_n$ into the concatenation of $\coprod_{x \in \Sigma^n} f(x)$, where x ranges over all strings $x \in \Sigma^n$ in a predetermined (such as lexicographical) order, and $\coprod$ is the concatenation operation on more than two strings.

By this bijection, $\psi(f_2, f_2, f_1)$ constructed from $f_1, f_2 \in \mathcal{H}_n$ via FTT's yields a function

$$S_{2n2^n} : \Sigma^{2n2^n} \to \Sigma^{2n2^{2n}}.$$

S_{2n2^n} maps a string $x = x_1 x_2$ where $x_1, x_2 \in \Sigma^{n2^n}$, into a string $y \in \Sigma^{2n2^{2n}}$ in the follow way:

$$S_{2n2^n}(x) = \Phi_{2n}(\psi(\Phi_n^{-1}(x_2), \Phi_n^{-1}(x_2), \Phi_n^{-1}(x_1))).$$

Now we describe concretely an algorithm G_n computing the function S_{2n2^n}. The algorithm follows a similar one in [Sch88]. (The reader is referred to Figure 4.2 before going into the details.) We write a string $x \in \Sigma^{2n2^n}$ as the concatenation of two strings $x_1, x_2 \in \Sigma^{n2^n}$, each of which is written as the concatenation of 2^n strings in Σ^n, i.e.,

$$x = x_1 x_2 = \coprod_{i \in \Sigma^n} x_{1,i} \coprod_{i \in \Sigma^n} x_{2,i},$$

where $x_{1,i}, x_{2,i} \in \Sigma^n$. Likewise, we write a string $y \in \Sigma^{2n2^{2n}}$ as the concatenation of 2^{2n} strings in Σ^{2n}, i.e., $y = \coprod_{i \in \Sigma^{2n}} y_i$, where $y_i \in \Sigma^{2n}$. For a string $y \in \Sigma^{2n}$, let $B_1(y)$ and $B_2(y)$ be the left and right half strings in Σ^n.

Algorithm $G_n(x)$

/* This algorithm outputs a $2n2^{2n}$-bit string y
on input a $2n2^n$-bit string $x = x_1 x_2 = \coprod_{i \in \Sigma^n} x_{1,i} \coprod_{i \in \Sigma^n} x_{2,i}$. */

(1) **For all** $i \in \Sigma^{2n}$, **let** $y_i^0 := i$;

(2) **For all** $i \in \Sigma^{2n}$ **do**

$\{\ w := y_i^0;\ u := x_{1,B_1(w)};$

$y_i^1 := (B_2(w) \oplus u, B_1(w))\ \ \};$

(3) **For** $j = 1, 2$ **do**

For all $i \in \Sigma^{2n}$ **do**

$\{\ w := y_i^j;\ u := x_{2,B_1(w)};$

$y_i^{j+1} := (B_2(w) \oplus u, B_1(w))\ \ \};$

(4) **Output** $y = \coprod_{i \in \Sigma^{2n}} y_i^3$.

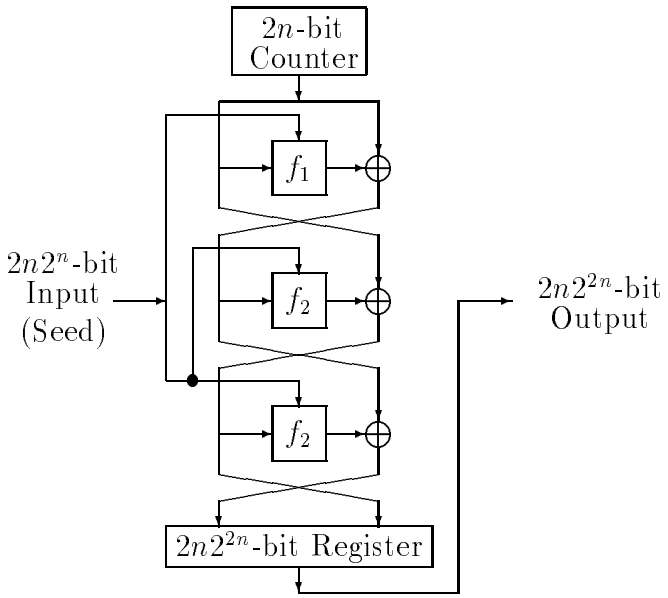


Figure 4.2: Algorithm G_n for Computing S_{2n2^n}

Let $S = \{S_{e(n)} | e(n) = 2n2^n, n \in \mathcal{N}\}$. From [Sch88] we know that the PSG S passes all statistical tests for strings which receive at most $2^{o(n)}$ bits as input. In our terms, this can be formally stated as follows.

Theorem 16 *The PSG* $S = \{S_{e(n)} | e(n) = 2n2^n, n \in \mathcal{N}\}$ *is locally* $(\infty, \mathcal{L}, \Upsilon)$*-secure where* $\mathcal{L}$ *is the infinite set of functions* L *in* n *with* $L(n) \leq 2^{o(n)}$ *and* Υ *that of the reciprocals of all polynomials in* n.

Proof: A local statistical test for strings $T^s = \{T_n^s \mid n \in \mathcal{N}\}$, where T_n^s has a $Q(n)(\leq 2^{o(n)})$-bit input, can be viewed as a statistical test for functions $T^f = \{T_n^f \mid n \in \mathcal{N}\}$,

where T_n^f has at most $Q(n) \leq 2^{o(n)}$ oracle gates that are evaluated using a function from $\mathcal{H}_{2n}$. Thus the theorem is true by FTT Lemma (Version 2) in Section 4.1. □

Applying our observation made in Section 4.1, this theorem can be translated into the following theorem.

Theorem 17 *Let $n = \lceil (\log t)^{1+\varepsilon} \rceil$, where $t \in \mathcal{N}$ and $\varepsilon \in (0, 1]$. Then the PSG*

$$S = \{S_{\bar{e}(t)} | \bar{e}(t) = e(n) = 2n2^n, n = \lceil (\log t)^{1+\varepsilon} \rceil, t \in \mathcal{N}\}$$

where $S_{\bar{e}(t)}$ maps an $\bar{e}(t) = e(n) = 2n2^n$-bit string into a $2n2^{2n}$-bit one, is locally $(\infty, \mathcal{L}, \Upsilon)$-secure where $\mathcal{L}$ is the infinite set of all polynomials in t, and Υ that of the reciprocals of all polynomials in t. That is to say, S is a robust PSG.

4.3.2 PSBC with Key-Expanding

Let $n = \lceil (\log t)^{1+\varepsilon} \rceil$, where $t \in \mathcal{N}$ and $\varepsilon \in (0, 1]$. Let Σ^{kn} be the plaintext/ciphertext spaces where $k = 2\ell$, $\ell \in \mathcal{N}$, and let ρ be an odd integer in $[1, k]$, s an integer with $s \geq k + 2$.

The enciphering algorithm consists of two parts: *the enciphering part* and *the key-expanding part* (Figure 4.3). The enciphering part, as PSBC, consists essentially of s rounds of Generalized Type-2 transformations. The key-expanding part is an algorithm G_m that computes a function $S_{\hat{e}(t)}$ from a robust PSG

$$S = \{S_{\hat{e}(t)} | \hat{e}(t) = 2m2^m, m = n\lceil \log n \rceil, n = \lceil (\log t)^{1+\varepsilon} \rceil, t \in \mathcal{N}\}$$

and it can expand a $2m2^m$-bit input string into a $2m2^{2m}$-bit output string.

The deciphering algorithm is obtained by

1. reversing the portion, which is used by the enciphering part, of the output of the key-expanding part and
2. changing the mapping (or wiring) representing $\mathrm{L}_{rot}^{(\rho)}$ to the mapping (or wiring) representing $\mathrm{R}_{rot}^{(\rho)}$.

The following theorem implies that the block cipher PSBC with key-expanding is secure against any polynomial size adversary. It can be proved by making some obvious modifications on the proof for Theorem 1 of [LR88].

Theorem 18 *Let $k = 2\ell$ where $\ell \in \mathcal{N}$, and ρ be an odd integer in $[1, k]$. Also let $t \in \mathcal{N}$, $\varepsilon \in (0, 1]$, $n = \lceil (\log t)^{1+\varepsilon} \rceil$ and*

$$S = \{S_{m(t)} | \hat{e}(t) = 2m2^m, m = n\lceil \log n \rceil, n = \lceil (\log t)^{1+\varepsilon} \rceil, t \in \mathcal{N}\}$$

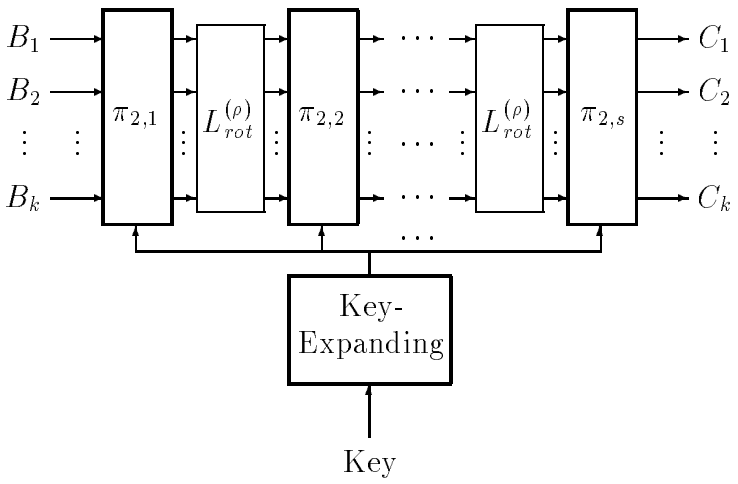


Figure 4.3: Enciphering Algorithm for PSBC with Key-Expanding

be the above constructed robust PSG. Assume that P and Q are polynomials in t and that C_{2n} is an oracle circuit with $Q(t)$ oracle gates. Then for any $r \in_R \mathcal{H}_{kn}$, for any $x \in_R \Sigma^{\hat{e}(t)}$, and for any $h_1, h_2, \ldots, h_{k+1}$ where $h_i = (f_{i,1}, f_{i,3}, \ldots, f_{i,k-1})$ and each $f_{i,j}$ corresponds to a distinct $n2^n (= 2^{o(m)})$-bit portion of the output of $S_{\hat{e}(t)}(x)$, we have

$$|\Pr\{C_{kn}[r] = 1\} - \Pr\{C_{kn}[\psi_2^{(\rho)}(h_{k+1}, \ldots, h_2, h_1)] = 1\}| < 1/P(t).$$

4.4 Practical Block Ciphers

PSBC with key-expanding requires still quasi-polynomial amount of memory to specify an enciphering/deciphering algorithm. In addition, the enciphering/deciphering part uses only an extremely small portion of the output of the key-expanding part.

Experience tells us that concatenating a number of transformations, each of which may not be so cryptographically robust, can produce a very robust one [Mey78]. This folklore has recently been proved to be correct by Luby and Rackoff. See Theorem 2 in the preliminary version of [LR88].

Along this guideline, we consider how to modify PSBC with key-expanding so that it is practically secure and can be implemented with current technology. We focus on the following three aspects: the size of a key, the sizes of n and k, and the rounds of transformations.

1. A key should be relatively short to make the cipher easy to be implemented. However to beat back the exhaustive search attack, the key should not be too short.