On the Construction of Pseudo-Random Permutations: Luby-Rackoff Revisited

(Extended Abstract)*

Moni Naor † Omer Reingold ‡

Abstract

Luby and Rackoff [21] showed a method for constructing a pseudo-random permutation from a pseudo-random function. The method is based on composing four (or three for weakened security) so called Feistel permutations, each of which requires the evaluation of a pseudo-random function. We reduce somewhat the complexity of the construction and simplify its proof of security by showing that two Feistel permutations are sufficient together with initial and final pair-wise independent permutations. The revised construction and proof provide a framework in which similar constructions may be brought up and their security can be easily proved. We demonstrate this by presenting some additional adjustments of the construction that achieve the following:

- Reduce the success probability of the adversary.
- Provide a construction of pseudo-random permutations with *large* input size using pseudo-random functions with *small* input size.
- Provide a construction of a pseudo-random permutation using a *single* pseudo-random function.

*A full version of this paper is available as *Theory of Cryptography Library: Record 96-11* at:
http://theory.lcs.mit.edu/~ tcryptol/homepage.html

†Incumbent of the Morris and Rose Goldman Career Development Chair, Dept. of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel. Research supported by grant no. 356/94 from the Israel Science Foundation administered by the Israeli Academy of Sciences. E-mail: naor@wisdom.weizmann.ac.il.

‡Dept. of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel. Part of this research was supported by a Clore Scholars award. E-mail: reingold@wisdom.weizmann.ac.il.

1 Introduction

Pseudo-random (**p.r.**) permutations, which were introduced by Luby and Rackoff [21], formalize the well established cryptographic notion of block ciphers. Block ciphers are private-key encryption schemes such that the encryption of every plaintext-block is a single ciphertext-block *of the same length*. Therefore we can think of the private key as determining a permutation on strings of the length of the block. A highly influential example of a block cipher is the Data Encryption Standard (DES) [27].

An important feature of block ciphers is that the plaintext and ciphertext are of the same length. This property saves on memory and prevents wasting communication bandwidth. Furthermore, it enables the easy incorporation of the encryption scheme into existing protocols or hardware components.

Luby and Rackoff defined the security of p.r. permutations in analogy to the different attacks considered in the context of block ciphers:

- P.r. permutations can be interpreted as block ciphers that are secure against an adaptive *chosen-plaintext attack*. Informally, this means that an (efficient) adversary, with access to the encryptions of messages of its choice, cannot tell apart those encryptions from the values of a truly random permutation.
- Strong p.r. permutations can be interpreted as block ciphers that are secure against an adaptive *chosen plaintext and ciphertext attack*. Here, the adversary has the additional power to ask for the decryption of ciphertexts of its choice.

P.r. permutations are closely related (both in definition and in their construction) to the earlier concept of p.r. functions which was defined by Goldreich, Goldwasser and Micali [14]. These are efficiently samplable and computable functions that are indistinguish-

able from random functions under all (efficient) black-box attacks (see Section 2 for a formal definition). P.r. functions play a major role in private-key cryptography and have many additional applications (for some of these applications, see [9, 15, 20]).

Luby and Rackoff [21] provided a construction of strong p.r. permutations, (**LR-Construction**) which was motivated by the structure of DES. The basic building block is the so called Feistel permutation (see Definition 2.1) based on a p.r. function defined by the key. Their construction consists of four rounds of Feistel permutations (or three rounds, for p.r. permutations) each round involves an application of a (different) p.r. function (see Figure 1.a for an illustration). The LR-Construction's main source of attraction is, most probably, its elegance.

Goldreich, Goldwasser and Micali [14] showed a construction of p.r. functions from p.r. generators [8, 34]. Thus, the construction of p.r. permutations reduces to the construction of p.r. generators. Recently a different construction of p.r. functions was introduced by Naor and Reingold [25]; this is a parallel construction based on a new primitive called a p.r. *synthesizer* that in particular can be constructed from any trapdoor permutation. This implies a parallel construction of p.r. permutations. Nevertheless, all known constructions of p.r. functions involve non-trivial (though of course polynomial time) computation, so it makes sense to attempt to minimize the number of invocations of p.r. functions.

Alongside cryptographic pseudo-randomness the last two decades saw the development of the notion of limited independence in various setting and formulations [2, 3, 10, 11, 19, 24, 33]. For a family of functions $\mathcal{F}$ to have some sort of (limited) independence means that if we consider the value of a function f, chosen uniformly at random from $\mathcal{F}$, at each point as a random variable (in the probability space defined by choosing f) then these random variables possess the promised independence property. Thus, a family of permutations on $\{0,1\}^n$ is pair-wise independent if for all $x \neq y$ the values of $f(x)$ and $f(y)$ are uniformly distributed over strings $(a,b) \in \{0,1\}^{2n}$ such that $a \neq b$. Functions of limited independence are typically much simpler to construct and easier to compute than (cryptographic) p.r. functions.

1.1 New Results and Organization

The goal of this paper is to provide a better understanding of the LR-Construction and as a result improve the construction in several respects. Our main observation is that the different rounds of the LR-Construction serve significantly different roles. We show that the first and last rounds can be replaced by pair-wise independent permutations and use this in order to :

1. Achieve an improvement in the computational complexity of the p.r. permutations – two applications of a p.r. function on n bits suffice for computing the value of a p.r. permutation on $2n$ bits at a given point (vs. four applications in the original LR-Construction).

2. Simplify the proof of security of the construction (especially in the case of strong p.r. permutations) and provide a framework for proving the security of similar constructions.

3. Derive generalizations of the construction that are of practical and theoretical interest. The proof of security for each one of the constructions is practically "free of charge" given the proof of security of the main construction.

The new construction is in fact a generalization of the original LR-Construction. Thus, the proof of security (Theorem 3.2) also applies to the original construction.

The paper is organized as follows: Section 2 reviews notations and definitions. Section 3 presents the main construction and proves its security. Section 4 provides a framework that enables us to relax and generalize the main construction. Section 5 provides a simple generalization of the main construction that significantly reduces the success probability of the distinguisher. Section 6 provides a second generalization of the main construction. This is a construction of a strong p.r. permutation on *large* blocks using p.r. functions on *small* blocks. Section 7 suggests directions for further research.

We omit in this version some of the proofs and discussions (see the full paper [26]). In particular, we omit the different relaxations of the main construction (using weaker and more efficient permutations instead of the pair-wise independent permutations and using a single p.r. function). A discussion on the connection of this paper to the constructions of k-wise δ-dependent permutations is omitted as well.

1.2 Related Work

The LR-Construction inspired a considerable amount of research. We try to refer to the more relevant (to this paper) part of these directions.

Several alternative proofs of the LR-Construction were presented over the years. Maurer [23] gives a proof of the three-round construction. His proof concentrates on the non-adaptive case, i.e., when the distinguisher has to specify all its queries in advance. A point worth noticing is that indistinguishability under non-adaptive attacks does not necessarily imply indistinguishability under adaptive attacks. For example, a random involution (an involution is a permutation which is the inverse of itself) and a random permutation are indistin-

guishable under non-adaptive attacks and can be distinguished using a very simple adaptive attack.A different approach toward the proof was described by Patarin [28] (this is the only published proof, we are aware of, for the LR-Construction of *strong* p.r. permutations; another proof was given by Koren [18]).

Other papers consider the security of possible variants of the construction. A significant portion of this research deals with the construction of p.r. permutations and strong p.r. permutations from a *single* p.r. function. Apparently, this line of research originated in the work of Schnorr [32]. This issue is explored in the full version [26].

Lucks [22] shows that a hash function can replace the p.r. function in the first round of the three-round LR-Construction. His proof is based on [23] and is motivated by his suggestion to use the LR-Construction when the input is divided into two *unequal* parts. Lucks left open the question of the construction of strong p.r. permutations.

Somewhat different questions were considered by Even and Mansour [12] and by Kilian and Rogaway [17]. Loosely speaking, the former construct several p.r. permutations from a single one, while the latter show how to make exhaustive key-search attacks more difficult. The background and related work concerning other relevant issues are discussed in the appropriate sections.

2 Preliminaries

In this section, the concepts of p.r. functions and p.r. permutations are briefly reviewed. A more thorough and formal treatment can be found in [13, 20]. In addition, some basic notations and definitions are introduced.

2.1 Notations

I_n denotes the set of all n-bit strings, $\{0,1\}^n$. F_n denotes the set of all $I_n \mapsto I_n$ functions and P_n denotes the set of all such permutations ($P_n \subset F_n$). Let x and y be two bit strings of equal length, then $x \oplus y$ denotes their bit-by-bit exclusive-or. For any $f, g \in F_n$ denote by $f \circ g$ their composition (i.e., $f \circ g(x) = f(g(x))$). For $x \in I_{2n}$, denote by $x_{|_L}$ the first (left) n bits of x and by $x_{|_R}$ the last (right) n bits of x.

Definition 2.1 (Feistel Permutations)
For any function $f \in F_n$, let $D_f \in P_{2n}$ be the permutation defined by $D_f(L,R) \stackrel{\text{def}}{=} (R, L \oplus f(R))$, where $|L| = |R| = n$.

Notice that Feistel permutations are as easy to invert as they are to compute. Therefore, the LR-Construction (and its different variants which are introduced in Sections 5 & 6) are easy to invert.

2.2 Pseudo-Randomness

Pseudo-randomness is fundamental to cryptography and, indeed, essential in order to perform such tasks as encryption, authentication and identification. Loosely speaking, p.r. distributions cannot be efficiently distinguished from the truly random distributions (usually, random here means uniform). However, the p.r. distributions have substantially smaller entropy than the truly random distributions and are efficiently samplable.

In the case of **p.r. (bit) generators**, which were introduced by Blum and Micali and Yao [8, 34], the p.r. distribution is of bit-sequences. The distribution is efficiently sampled using a, relatively small, truly random bit-sequence (the seed). Hastad, Impagliazzo, Levin and Luby [16] showed how to construct a p.r. generator from any one-way function (informally, a function is one-way if it is easy to compute its value but hard to invert it).

P.r. function ensembles (PFE), which were introduced by Goldreich, Goldwasser and Micali [14], are distributions of functions. These distributions are indistinguishable from the uniform distribution under all (polynomially-bounded) black-box attacks (i.e. the distinguisher can only access the function by specifying inputs and getting the value of the function on these inputs). Goldreich, Goldwasser and Micali provided a construction of such functions based on the existence of p.r. generators.

Luby and Rackoff [21] define **p.r. permutation ensembles (PPE)** to be distributions of permutations that are indistinguishable from the uniform distribution to an efficient observer (that, again, has access to the value of the permutation at points of its choice). In addition, they consider a stronger notion of pseudo-randomness which they call *super p.r. permutation generators*. Here the distinguisher can also access the inverse permutation at points of its choice. Following [13] we use the term **strong p.r. permutation ensembles (SPPE)** instead.

Luby and Rackoff provided a simple construction of PPE and SPPE *(LR-Construction)* which is the focus of this work. Their construction is based on a basic compound of the structure of DES [27], namely, the compositions of several Feistel-permutations. Their definition of the PPE (resp. SPPE) is $D_{f_3} \circ D_{f_2} \circ D_{f_1}$ (resp. $D_{f_4} \circ D_{f_3} \circ D_{f_2} \circ D_{f_1}$) where all f_is are independent p.r. functions and D_{f_i} as in Definition 2.1 (see Figure 1.a for an illustration).

2.2.1 Definitions

A *function ensemble* is a sequence $H = \{H_n\}_{n \in \mathbb{N}}$ such that H_n is a distribution over F_n, H is the *uniform function ensemble* if H_n is uniformly distributed over F_n. A *permutation ensemble* is a sequence $H = \{H_n\}_{n \in \mathbb{N}}$ such that H_n is a distribution over P_n, H is the *uniform permutation ensemble* if H_n is uniformly distributed over P_n.

A function (or permutation) ensemble, $H = \{H_n\}_{n \in \mathbb{N}}$, is *efficiently computable* if the distribution H_n can be sampled efficiently (in probabilistic polynomial-

time) and the functions in H_n can be computed efficiently.

We would like to consider efficiently computable function (or permutation) ensembles that cannot be efficiently distinguished from the uniform ensemble. In our setting, the distinguisher is an oracle machine that can make queries to a length preserving function (or functions) and outputs a single bit. We assume that on input 1^n the oracle machine makes only n-bit long queries, n also serves as the security parameter. The discussion of this paper is independent of whether we interpret an oracle machine as a Turing-machine with a special oracle-tape or as a circuit-family with special oracle-gates.

Let M be an oracle machine, let f be a function in F_n and H_n a distribution over F_n. Denote by $M^f(1^n)$ the distribution of M's output when its queries are answered by f and denote by $M^{H_n}(1^n)$ the distribution $M^f(1^n)$, where f is distributed according to H_n. We would also like to consider oracle machines with access both to a permutation and to its inverse. Let M be such a machine, let f be a permutation in P_n and H_n a distribution over P_n. Denote by $M^{f,f^{-1}}(1^n)$ the distribution of M's output when its queries are answered by f and f^{-1} and denote by $M^{H_n,H_n^{-1}}(1^n)$ the distribution $M^{f,f^{-1}}(1^n)$, where f is distributed according to H_n.

Definition 2.2 (advantage) *Let M be an oracle machine and let $H = \{H_n\}_{n\in\mathbb{N}}$ and $\tilde{H} = \{\tilde{H}_n\}_{n\in\mathbb{N}}$ be two function (or permutation) ensembles. We call the function*

$$|\Pr[M^{H_n}(1^n) = 1] - \Pr[M^{\tilde{H}_n}(1^n) = 1]|$$

the advantage *M achieves in distinguishing between H and $\tilde{H}$.*

Let M be an oracle machine and let $H = \{H_n\}_{n\in\mathbb{N}}$ and $\tilde{H} = \{\tilde{H}_n\}_{n\in\mathbb{N}}$ be two permutation ensembles. We call the function

$$|\Pr[M^{H_n,H_n^{-1}}(1^n) = 1] - \Pr[M^{\tilde{H}_n,\tilde{H}_n^{-1}}(1^n) = 1]|$$

the advantage *M achieves in distinguishing between $\langle H, H^{-1}\rangle$ and $\langle \tilde{H}, \tilde{H}^{-1}\rangle$.*

We say that M distinguishes between H and $\tilde{H}$ (resp. $\langle H, H^{-1}\rangle$ and $\langle \tilde{H}, \tilde{H}^{-1}\rangle$) with advantage $\epsilon = \epsilon(n)$ if for infinitely many n's the advantage M achieves in distinguishing between H and $\tilde{H}$ (resp. $\langle H, H^{-1}\rangle$ and $\langle \tilde{H}, \tilde{H}^{-1}\rangle$) is at least $\epsilon(n)$.

Definition 2.3 (negligible functions) *A function $h : \mathbb{N} \mapsto \mathbb{N}$ is* negligible *if for every constant $c > 0$ and all sufficiently large n's, $h(n) < \frac{1}{n^c}$*

Definition 2.4 (PFE) *Let $H = \{H_n\}_{n\in\mathbb{N}}$ be an efficiently computable* function *ensemble and let $R = \{R_n\}_{n\in\mathbb{N}}$ be the uniform function ensemble. H is a* p.r. function ensemble *if for every efficient oracle-machine M, the advantage M has in distinguishing between H and R is negligible.*

Definition 2.5 (PPE) *Let $H = \{H_n\}_{n\in\mathbb{N}}$ be an efficiently computable* permutation *ensemble and let $R = \{R_n\}_{n\in\mathbb{N}}$ be the uniform permutation ensemble. H is a* p.r. permutation ensemble *if for every efficient oracle-machine M, the advantage M has in distinguishing between H and R is negligible.*

Definition 2.6 (SPPE) *Let $H = \{H_n\}_{n\in\mathbb{N}}$ be an efficiently computable permutation ensemble and let $R = \{R_n\}_{n\in\mathbb{N}}$ be the uniform permutation ensemble. H is a* strong p.r. permutation ensemble *if for every efficient oracle-machine M, the advantage M has in distinguishing between $\langle H, H^{-1}\rangle$ and $\langle R, R^{-1}\rangle$ is negligible.*

Remark 2.1 *We use the phrase "f is a p.r. function" as an abbreviation for "f is distributed according to a p.r. function ensemble" and similarly for "f is a p.r. permutation" and "f is a strong p.r. permutation"*

2.3 k-Wise Independent Functions and Permutations

The notions of k-wise independent functions and k-wise "almost" independent functions [2, 3, 10, 11, 19, 24, 33] (under several different formulations) play a major role in contemporary computer science. These are distributions of functions such that their value on any given k inputs is uniformly or "almost" uniformly distributed. Several constructions of such functions and a large variety of applications were suggested over the years.

As shown in Section 3, pair-wise independent permutations can replace the first and fourth rounds of the LR-Construction. We briefly review the definitions of pair-wise independent permutations and functions:

Definition 2.7 *Let A and B be two sets and F a distribution of $A \mapsto B$ functions. F is pair-wise independent if for every two members $x_1 \neq x_2$ of A, $\langle f(x_1), f(x_2)\rangle$ is uniformly distributed over B^2.*

This definitions is naturally extended to permutations:

Definition 2.8 *Let A be a set and F a distribution of permutations over A. F is pair-wise independent if for every two members $x_1 \neq x_2$ of A, $\langle f(x_1), f(x_2)\rangle$ is uniformly distributed over pairs of different elements of A.*

Let A be a finite field then the permutation $f_{a,b}(x) \stackrel{\text{def}}{=} a \cdot x + b$, where $a \neq 0, b \in A$ are uniformly distributed, is pair-wise independent. Thus, there are pair-wise independent permutations over I_n (the permutations $f_{a,b}$ with operations over $GF(2^n)$). In the full version, it is shown that we can use even more efficient functions and permutations in our construction.

There is another connection between this paper and k-wise independence: In contrast with the case of pair-wise independent permutations, we are not aware of any

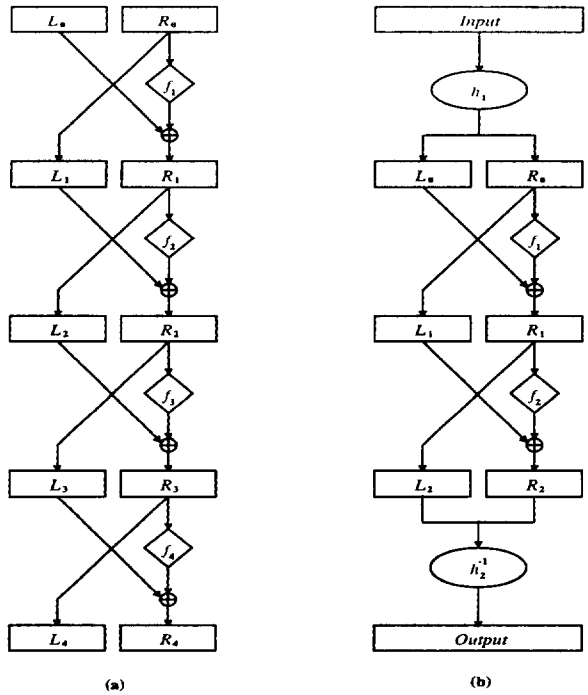

Figure 1: Constructions of SPPE: (a) The original LR-Construction (b) The revised Construction. In (a) and (b): $\forall i \geq 1$, $L_i = R_{i-1}$ and $R_i = L_{i-1} \oplus f_i(R_{i-1})$. In (b): $\langle L_0, R_0 \rangle = h_1(Input)$ and $Output = h_2^{-1}(\langle L_2, R_2 \rangle)$.

"good" constructions of k-wise δ-dependent permutations for general k and δ. The different variants of the LR-Construction offer a partial solution to this problem ("partial" because of the bounded values of δ that can be achieved). For example, using k-wise δ'-dependent functions on n bits instead of p.r. functions in the original LR-Construction yields k-wise δ-dependent permutation on $2n$ bits (for $\delta = O(k^2/2^n + \delta')$). In [26], we analyze the different constructions of this paper as constructions of k-wise δ-dependent permutations.

3 Construction of PPE and SPPE

As mentioned in the introduction, a principle observation of this paper is that the different rounds of the LR-Construction serve significantly different roles. To illustrate this point, consider two rounds of the construction. Namely, $E = D_{f_2} \circ D_{f_1}$, where $f_1, f_2 \in F_n$ are two independently chosen p.r. functions. It is not hard to verify that E is computationally indistinguishable from a random permutation to any efficient algorithm that has access to pairs $\{\langle x_i, E(x_i) \rangle\}_{i=1}^m$, where the sequence $\{x_i\}_{i=1}^m$ is uniformly distributed. Nevertheless, as Luby and Rackoff showed, E can be easily distinguished from a random permutation by an algorithm that gets to see the value of E or E^{-1} on inputs of its choice.

We think of the second and third rounds of the LR-Construction as the two-round construction E, described above and show that the role of the first and fourth rounds is to prevent the distinguisher from directly choosing the inputs of E and E^{-1}. As we shall see, this goal can also be achieved with "combinatorial" constructions (e.g., pair-wise independent permutations), rather than "cryptographic" (i.e., p.r. functions). In particular, the LR-Construction remains secure when the first and fourth rounds are replaced with pair-wise independent permutations

3.1 Construction and Main Result

Definition 3.1 *For any $f_1, f_2 \in F_n$ and $h_1, h_2 \in P_{2n}$, define*

$$W(h_1, f_1, f_2) \stackrel{\text{def}}{=} D_{f_2} \circ D_{f_1} \circ h_1$$

and

$$S(h_1, f_1, f_2, h_2) \stackrel{\text{def}}{=} h_2^{-1} \circ D_{f_2} \circ D_{f_1} \circ h_1$$

Theorem 3.1 *Let $h_1, h_2 \in P_{2n}$ be pair-wise independent permutations (similarly to Remark 2.1 this is an abbreviation for "distributed according to a pair-wise independent permutation ensemble") and let $f_1, f_2 \in F_n$ be p.r. functions; h_1, h_2, f_1 and f_2 are independently chosen. Then, $W = W(h_1, f_1, f_2)$ is a p.r. permutation and $S = S(h_1, f_1, f_2, h_2)$ is a strong p.r. permutation (W and S as in Definition 3.1).*

Furthermore, assume that no efficient oracle-machine that makes at most $m = m(n)$ queries, distinguishes between the p.r. functions and random functions with advantage $\epsilon = \epsilon(n)$ (see Definition 2.2). Then, no efficient oracle-machine that makes at most m queries to W (resp. S and S^{-1}) distinguishes W (resp. S) from a random permutation with advantage $2\epsilon + \frac{m^2}{2^n} + \frac{m^2}{2^{2n}}$

Remark 3.1 *The conditions of Theorem 3.1 are meant to simplify the exposition of the theorem and of its proof. These conditions can be relaxed, as discussed in [26]. The main points are the following:*

1. *A single p.r. function f can replace both f_1 and f_2*
2. *h_1 and h_2 may obey weaker requirements than pair-wise independence. For example, it is enough that for every $x \neq y$ both $\Pr[h_1(x)_{|R} = h_1(y)_{|R}] \leq 2^{-n}$ and $\Pr[h_2(x)_{|L} = h_2(y)_{|L}] \leq 2^{-n}$.*

An important consequence of the second relaxation is that the original LR-Construction is a special case of the revised construction.

3.2 Proof of Security

We now prove the security of the SPPE-construction; the proof of security for the PPE-construction is very similar (and, in fact, a bit simpler). As with the original LR-Construction, the main task is to prove that the permutations are p.r. when f_1 and f_2 are truly random (instead of p.r.).

Theorem 3.2 *Let $h_1, h_2 \in P_{2n}$ be pair-wise independent permutations and let $f_1, f_2 \in F_n$ be random functions. Define $S = S(h_1, f_1, f_2, h_2)$ (as in Definition 3.1) and let $R \in P_{2n}$ be a random permutation. Then, for any oracle machine M (not necessarily an efficient one) that makes at most m queries:*

$$|\Pr[M^{S,S^{-1}}(1^{2n}) = 1] - \Pr[M^{R,R^{-1}}(1^{2n}) = 1]| \leq \frac{m^2}{2^n} + \frac{m^2}{2^{2n}}$$

Theorem 3.1 follows easily from Theorem 3.2. In order to prove Theorem 3.2, we introduce some additional notations. Let G denote the permutation that is accessible to the machine M (G is either S or R). There are two types of queries M can make: either $(+, x)$ which denotes the query "what is $G(x)$?" or $(-, y)$ which denotes the query "what is $G^{-1}(y)$?". For the ith query M makes, define the query-answer pair $\langle x_i, y_i \rangle \in I_{2n} \times I_{2n}$, where either M's query was $(+, x_i)$ and the answer it got was y_i or M's query was $(-, y_i)$ and the answer it got was x_i. We assume that M makes exactly m queries and refer to the sequence $\{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$ of all these pairs as the *transcript* (of M's computation).

Notice that no limitations were imposed on the computational power of M. Therefore, M can be assumed to be deterministic (we can always fix the random tape that maximizes the advantage M achieves). This assumption implies that for every $1 \leq i \leq m$ the ith query of M is fully determined by the first $i-1$ query-answer pairs. Thus, for every i it can be determined from the transcript whether the ith query was $(+, x_i)$ or $(-, y_i)$. We also get that M's output is a (deterministic) function of its transcript. Denote by $C_M[\{\langle x_1, y_1 \rangle, ..., \langle x_{i-1}, y_{i-1} \rangle\}]$ the ith query of M as a function of the previous query-answer pairs and denote by $C_M[\{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}]$ the output of M as a function of its transcript.

Definition 3.2 *Let $\sigma = \{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$ be a sequence such that $\forall 1 \leq i \leq m$, $\langle x_i, y_i \rangle \in I_{2n} \times I_{2n}$. Then, σ is a* possible M-transcript *if for every $1 \leq i \leq m$*

$$C_M[\{\langle x_1, y_1 \rangle, ..., \langle x_{i-1}, y_{i-1} \rangle\}] \in \{(+, x_i), (-, y_i)\}$$

Let us consider yet another distribution on the answers to M's queries (which, in turn, induces another distribution on the possible M-transcripts). Consider a random process $\tilde{R}$ that on every query of M answers with a uniformly chosen $2n$-bit string unless the answer is implied by a previous query-answer pair (i.e., M's query is $(+, x)$ or $(-, y)$ and $\langle x, y \rangle$ is a previous query-answer pair). It is possible that $\tilde{R}$ provides answers that are not consistent with *any* permutation; that is, we can have two query-answer pairs of the form $\langle x_1, y \rangle$ and $\langle x_2, y \rangle$ for $x_1 \neq x_2$ or $\langle x, y_1 \rangle$ and $\langle x, y_2 \rangle$ for $y_1 \neq y_2$. In this case call the transcript *inconsistent*, otherwise, the transcript is called *consistent*.

We first show (in Proposition 3.3) that the advantage M might have in distinguishing between the process $\tilde{R}$ and the random permutation R is small. The reason is that as long as $\tilde{R}$ answers consistently (which happens with good probability) it "behaves" exactly as a random permutation. In order to formalize this, we consider the different distributions on the transcript of M (induced by the different distributions on the answers it gets).

Definition 3.3 *Let T_S, T_R and $T_{\tilde{R}}$ be the random variables such that T_S is the transcript of M when its queries are answered by S, T_R is the transcript of M when its queries are answered by R and $T_{\tilde{R}}$ is the transcript of M when its queries are answered by $\tilde{R}$.*

Notice that by these definitions (and by our assumptions) $M^{S,S^{-1}}(1^{2n}) = C_M(T_S)$ (are the same random variables) and $M^{R,R^{-1}}(1^{2n}) = C_M(T_R)$.

Proposition 3.3

$$\begin{aligned} & |\Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1] - \Pr_{R}[C_M(T_R) = 1]| \\ \leq\ & \Pr_{\tilde{R}}[T_{\tilde{R}} \text{ is inconsistent}] \\ \leq\ & \frac{m^2}{2^{2n+1}} \end{aligned}$$

It remains to bound the advantage M might have in distinguishing between $T_{\tilde{R}}$ and T_S. The intuition is that for every possible and consistent M-transcript σ unless some "bad" and "rare" event on the choice of h_1 and h_2 (as in the definition of S) happens, the probability that $T_S = \sigma$ is exactly the same as the probability that $T_{\tilde{R}} = \sigma$. We now formally define this event (Definition 3.4) and bound its probability (Proposition 3.4).

We can assume that for any possible M-transcript, $\sigma = \{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$, that is *consistent* we have that for $i \neq j$ both $x_i \neq x_j$ and $y_i \neq y_j$ (this means that M never asks a query if its answer is determined by a previous query-answer pair).

Definition 3.4 *For every specific choice of pair-wise independent permutations $h_1, h_2 \in P_{2n}$ (in the definition of S) define $BAD(h_1, h_2)$ to be the set of all possible and consistent M-transcripts, $\sigma = \{\langle x_1, y_1 \rangle, \ldots, \langle x_m, y_m \rangle\}$, satisfying that $\exists 1 \leq i < j \leq m$ such that:*

$$h_1(x_i)_{|_R} = h_1(x_j)_{|_R} \text{ or } h_2(y_i)_{|_L} = h_2(y_j)_{|_L}$$

Proposition 3.4 *Let $h_1, h_2 \in P_{2n}$ be pair-wise independent permutations then for any possible and consistent M-transcript $\sigma = \{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$ we have that:*

$$\Pr_{h_1, h_2}[\sigma \in BAD(h_1, h_2)] < \frac{m^2}{2^n}$$

The key lemma for proving Theorem 3.2 is:

Lemma 3.5 *Let $\sigma = \{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$ be any possible and consistent M-transcript, then*

$$\Pr_{S}[T_S = \sigma \mid \sigma \notin BAD(h_1, h_2)] = \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma]$$

Proof. Since σ is a possible M-transcript, we have that $T_{\tilde{R}} = \sigma$ iff for all $1 \leq i \leq m$, the ith answer $\tilde{R}$ gives is y_i in the case that $C_M[\{\langle x_1, y_1\rangle, ..., \langle x_{i-1}, y_{i-1}\rangle\}] = (+, x_i)$ and otherwise its ith answer is x_i. By our assumptions and the definition of $\tilde{R}$, given that $\tilde{R}$ answered "correctly" on each one of the first $i-1$ queries its ith answer is an independent and uniform $2n$-bit string. Therefore,

$$\Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma] = 2^{-2nm}$$

Since σ is a possible M-transcript we have that $T_S = \sigma$ iff for all $1 \leq i \leq m$, $y_i = S(x_i)$. Consider any specific choice of permutations h_1 and h_2 (for which $S = S(h_1, f_1, f_2, h_2)$) such that $\sigma \notin \mathrm{BAD}(h_1, h_2)$. Let $(L_i^0, R_i^0) = h_1(x_i)$ and $(L_i^2, R_i^2) = h_2(y_i)$. By the definition of S, we get that:

$$y_i = S(x_i) \iff f_1(R_i^0) = L_i^0 \oplus L_i^2 \text{ and } f_2(L_i^2) = R_i^0 \oplus R_i^2$$

For every $1 \leq i < j \leq m$ both $R_i^0 \neq R_j^0$ and $L_i^2 \neq L_j^2$ (otherwise $\sigma \in \mathrm{BAD}(h_1, h_2)$). Therefore, since f_1 and f_2 are random, we have that for every choice of h_1 and h_2 such that $\sigma \notin \mathrm{BAD}(h_1, h_2)$ the probability that $T_S = \sigma$ is exactly 2^{-2nm}. We can conclude:

$$\Pr_S[T_S = \sigma \mid \sigma \notin \mathrm{BAD}(h_1, h_2)] = 2^{-2nm}$$

which complete the proof of the lemma. □

Proof. **(of Theorem 3.2)** Let Γ be the set of all possible and consistent M-transcripts σ such that $M(\sigma) = 1$.

$$\begin{aligned}
&\left|\Pr_S[C_M(T_S) = 1] - \Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1]\right| \\
\leq\ &\left|\sum_{\sigma\in\Gamma}(\Pr_S[T_S = \sigma] - \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma])\right| \\
&+ \Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is inconsistent}] \\
\leq\ &\sum_{\sigma\in\Gamma}\left|\Pr_S[T_S = \sigma | \sigma \notin \mathrm{BAD}(h_1, h_2)] - \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma]\right| \cdot \\
&\Pr_{h_1,h_2}[\sigma \notin \mathrm{BAD}(h_1, h_2)] \qquad (1) \\
+\ &\left|\sum_{\sigma\in\Gamma}(\Pr_S[T_S = \sigma | \sigma \in \mathrm{BAD}(h_1, h_2)] - \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma]) \cdot \right. \\
&\left.\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}(h_1, h_2)]\right| \qquad (2) \\
+\ &\Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is inconsistent}] \qquad (3)
\end{aligned}$$

By Proposition 3.3, we get that that (3) $< \frac{m^2}{2^{2n+1}}$ and by Lemma 3.5 that (1) $= 0$. Using Proposition 3.4, it is not hard to show that (2) $< \frac{m^2}{2^n}$. Thus, we conclude:

$$|\Pr_S[C_M(T_S) = 1] - \Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1]| < \frac{m^2}{2^n} + \frac{m^2}{2^{2n+1}}$$

Using Proposition 3.3 we can complete the proof. □

4 The Framework

The construction of Section 3 can be relaxed and generalized in several ways. The different p.r. permutations obtained share a similar structure and almost identical proof of security. In this section we examine the proof of Theorem 3.2 in a more abstract manner. Our goal is to establish a framework for proving (almost) all the constructions of this paper and to suggest a way for designing and proving additional constructions.

Our framework deals with constructions of a p.r. permutation S on ℓ bits which is the composition of three permutations: $S \equiv h_2^{-1} \circ E \circ h_1$. In general, h_1 and h_2^{-1} are "lightweight" and E is where most of the work is done. E is constructed from p.r. functions and for the purpose of the analysis we assume (as in Theorem 3.2) that these functions are truly random. In Section 3, for example, $\ell = 2n$, h_1 and h_2 are chosen as pair-wise independent permutations and $E \equiv D_{f_2} \circ D_{f_1}$ for random $f_1, f_2 \in F_n$.

The framework starts with E which may be easily distinguished from a truly random permutation and transforms it via h_1 and h_2 into a p.r. permutation. The property E should have is that for almost every sequence, $\{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$, the probability that $\forall i$, $y_i = E(x_i)$ is "close" to what we have for a truly random permutation: Call a sequence, $\{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$, E-Good if $\Pr_E[\forall i,\ y_i = E(x_i)] = 2^{-l \cdot m}$. We assume that apart from some "rare" sequences all others are E-Good. Loosely speaking, the role of h_1 and h_2 is to ensure that under any (adaptive chosen plaintext and ciphertext) attack on S the inputs and outputs of E form an E-Good sequence with a very high probability.

For the exact properties needed from the distributions on h_1, h_2 and E, we shall try to follow the statement and proof of Theorem 3.2. The goal is to show that S is indistinguishable from a truly random permutation R on ℓ bits. Specifically, that for some small ϵ (whose choice will be explained hereafter), for any oracle machine M (not necessarily an efficient one) that makes at most m queries:

$$|\Pr[M^{S,S^{-1}}(1^\ell) = 1] - \Pr[M^{R,R^{-1}}(1^\ell) = 1]| \leq \epsilon + \frac{m^2}{2^\ell}.$$

Let the notions of query-answer pair, a transcript, the function C_M, a possible M-transcript, the random process $\tilde{R}$, a consistent transcript and the different random variables T_S, T_R and $T_{\tilde{R}}$ be as in the proof of Theorem 3.2. Proposition 3.3 still holds. The heart of applying the framework is in specifying the "bad" M-transcripts for given h_1 and h_2. This set $\mathrm{BAD}_E(h_1, h_2)$ replaces $\mathrm{BAD}(h_1, h_2)$ in Definition 3.4 and in the rest of the proof. It contains possible and consistent M-transcripts and should have the property that any $\{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$ *not in* $\mathrm{BAD}_E(h_1, h_2)$ satisfies that $\{\langle h_1(x_1), h_2(y_1)\rangle, \ldots, \langle h_1(x_m), h_2(y_m)\rangle\}$ is

E-Good. Note that Definition 3.4 is indeed a special case of the above and also that, by this property,

$$\Pr_S[T_S = \sigma \mid \sigma \notin \mathrm{BAD}_E(h_1, h_2)] = 2^{-\ell \cdot m}$$

This implies that Lemma 3.5 where $\mathrm{BAD}(h_1, h_2)$ is replaced with $\mathrm{BAD}_E(h_1, h_2)$ is true:

Lemma 4.1 *Let $\sigma = \{\langle x_1, y_1\rangle, ..., \langle x_m, y_m\rangle\}$ be any possible and consistent M-transcript, then*

$$\Pr_S[T_S = \sigma \mid \sigma \notin BAD_E(h_1, h_2)] = \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma].$$

For $\mathrm{BAD}_E(h_1, h_2)$ to be useful we must have that

$$\Pr_{h_1, h_2}[\sigma \in \mathrm{BAD}_E(h_1, h_2)] \leq \epsilon \qquad (1)$$

and this substitutes Proposition 3.4. This is the only place in the proof[1] where we use the definition of ϵ and the definition of the distributions of h_1 and h_2. Applying (1) and Lemma 4.1 as in the proof of Theorem 3.2 we conclude:

Theorem 4.2 *Let h_1, h_2, E be distributed over permutations in P_ℓ, let $S \equiv h_2^{-1} \circ E \circ h_1$ and let $R \in P_\ell$ be a random permutation. Suppose that $BAD_E(h_1, h_2)$ is as above and ϵ satisfies (1). Then, for any oracle machine M (not necessarily an efficient one) that makes at most m queries:*

$$|\Pr[M^{S,S^{-1}}(1^\ell) = 1] - \Pr[M^{R,R^{-1}}(1^\ell) = 1]| \leq \epsilon + \frac{m^2}{2^\ell}$$

To summarize, the major point in proving the security of the different constructions is to define the set $\mathrm{BAD}_E(h_1, h_2)$ such that for any possible and consistent M-transcript, σ, both $\Pr_S[T_S = \sigma \mid \sigma \notin \mathrm{BAD}_E(h_1, h_2)] = 2^{-\ell \cdot m}$ and $\Pr_{h_1, h_2}[\sigma \in \mathrm{BAD}_E(h_1, h_2)] \leq \epsilon$ (for the specific ϵ in the claim we are proving). This suggests that the critical step for designing a p.r. permutation, using the framework described in this section, is to come up with a permutation E such that the set of E-Good sequences is "large enough" and "nice enough". Note that to meet this end one can use different or more general definitions of an E-Good sequence with only minor changes to the proof (as is the case for the permutation $\hat{S}$ in Section 6).

5 Reducing the Distinguishing Probability

There are various circumstances where it is desirable to have a p.r. permutation on relatively *few* bits (say 128). This is especially true when we want to minimize the size of the hardware-circuit that implements the permutation or the communication bandwidth with the (hardware or software) component that computes the permutation.

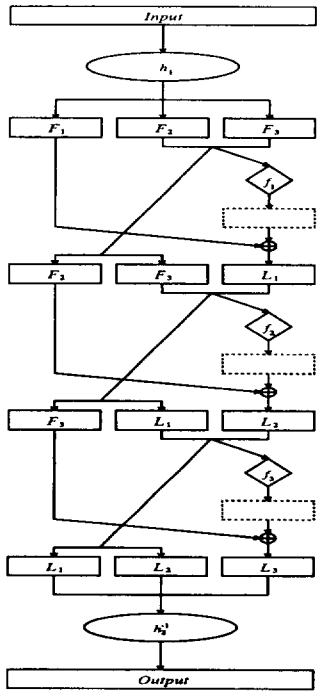

Figure 2: Construction of strong p.r. permutations with reduced distinguishing probability using $t + 2$ rounds (here $t = 3$). Recall, $f_i : I_{(1-1/t)\ell} \mapsto I_{\ell/t}$ (here $f_i : I_{2\ell/3} \mapsto I_{\ell/3}$).

Let F be a p.r. permutation on ℓ bits (note that $n = \ell/2$ in Section 3) constructed from truly random functions (on $\ell/2$ bits) using the LR-Construction. As shown by Patarin [29], F can be distinguished (with constant probability) from a random permutation using $O(2^{\ell/4})$ queries (which means that the analysis of the LR-Construction, where the distinguishing probability for m queries is $O(\frac{m^2}{2^{\ell/2}})$, is tight). Therefore, the LR-Construction on ℓ bits can only be used if $2^{\ell/4}$ is large enough to bound the number of queries in the attack on the block cipher.

In this section, a simple generalization of the construction of Section 3 is presented. Using this construction, the adversary's probability of distinguishing between the p.r. and random permutations can be reduced to roughly $\frac{t}{2} \cdot \frac{m^2}{2^{(1-1/t)\ell}}$ for every integer $2 \leq t \leq \ell$ (for $t = 2$ we get the original construction). To achieve this security $t+2$ permutations are composed. The initial and final are pair-wise independent permutations, the rest are (generalized) Feistel permutations defined by $I_{(1-1/t)\ell} \mapsto I_{\ell/t}$ random (or p.r.) functions.

Patarin [30] shows that if we take six rounds of the LR-Construction (instead of three or four), then the resulting permutation cannot be distinguished from

[1] As demonstrated in [26], there is actually a tradeoff between reducing the requirements from h_1 and h_2 and having a somewhat larger value of ϵ.

a random permutation with advantage better than $\frac{5m^3}{2^\ell}$ (improving [29]). This means that distinguishing the six-round construction from a truly random permutation (with constant probability) requires at least $\Omega(2^{\ell/3})$ queries. The bound we achieve in this section ($\Omega(2^{(1-1/t)\cdot\ell/2})$) is better (for any $t \geq 4$). Note that our construction uses p.r. functions with larger input size, which might be a disadvantage for some applications. Aiello and Venkatesan [1] show a construction of p.r. *functions* on ℓ bits from p.r. functions on $\ell/2$ bits. When using truly random functions in their construction, distinguishing the function they get from a truly random function (with constant probability) requires $\Omega(2^{\ell/2})$ queries.

To describe our generalized constructions we first extend Feistel permutations to deal with the case where the underlying functions have arbitrary input and output lengths (instead of length preserving functions as in Definition 2.1). Note that using such "unbalanced" Feistel permutations was previously suggested in [4, 22, 31].

Definition 5.1 (Generalized Feistel Permutations) *For any integers $0 < s < \ell$ and any function $f : I_{\ell-s} \mapsto I_s$ let $D_f \in P_\ell$ be the permutation defined by $D_f(L,R) \stackrel{\text{def}}{=} (R, L \oplus f(R))$, where $|L| = s$ and $|R| = \ell - s$*

We can now define the revised construction and consider its security. These are simple generalizations of the construction in Section 3 and of its proof of security. For lack of space, we only describe the construction with truly random functions.

Definition 5.2 ($t+2$-Round Construction) *For $\ell = s \cdot t$ (see [26] for the general case where $\ell = s \cdot t + r$), for any $h_1, h_2 \in P_\ell$ and $f_1, f_2, \ldots, f_t : I_{\ell-s} \mapsto I_s$ define*

$$W(h_1, f_1, f_2 \ldots, f_t) \stackrel{\text{def}}{=} D_{f_t} \circ D_{f_{t-1}} \circ \ldots \circ D_{f_1} \circ h_1$$

and

$$S(h_1, f_1, f_2, \ldots, f_t, h_2) \stackrel{\text{def}}{=} h_2^{-1} \circ D_{f_t} \circ D_{f_{t-1}} \circ \ldots \circ D_{f_1} \circ h_1$$

Theorem 5.1 *Let S be as in Definition 5.2, where h_1 and h_2 are pair-wise independent permutations and $f_1, f_2, \ldots, f_t$ are random functions and let $R \in P_\ell$ be a random permutation. Then, for any oracle machine M (not necessarily an efficient one) that makes at most m queries:*

$$\begin{aligned} & |\Pr[M^{S,S^{-1}}(1^\ell) = 1] - \Pr[M^{R,R^{-1}}(1^\ell) = 1]| \\ \leq\ & \frac{t}{2} \cdot \frac{m^2}{2^{\ell - \lceil \ell/t \rceil}} + \frac{m^2}{2^\ell} \end{aligned}$$

The proof of Theorem 5.1 follows the framework described in Section 4.

Remark 5.1 *The construction of this section achieves a substantial improvement in security over the construction in Section 3 even for a small constant $t > 2$ (that is, with a few additional applications of the p.r. functions). Nevertheless, it might be useful for some applications to take a larger value of t. Choosing $t = \ell$ reduces the advantage the distinguisher may achieve to roughly $\frac{\ell \cdot m^2}{2^\ell}$.*

6 SPPE on Large Blocks Using PFE or PPE on Small Blocks

Consider the application of p.r. permutations to encryption, i.e., using $f(M)$ in order to encrypt a message M, where f is a p.r. permutation. Assume also that we want to use DES for this purpose. We now have the following problem: while DES works on fixed and relatively small length strings, we need a permutation on $|M|$-bit long strings, where the length of the message, $|M|$, may be large and may vary between different messages.

This problem is not restricted to the usage of DES (though the fact that DES was designed for hardware implementation contributes to it). Usually, a direct construction of p.r. permutations or p.r. functions (if we want to employ the LR-Construction) with large input size is expensive. Therefore, we would like a way to construct p.r. permutations (or functions) on *large blocks* from p.r. permutations (or functions) on *small blocks*.

Several such constructions were suggested in the context of DES (see e.g. [9] for the different modes of operation for DES). The simplest, known as the electronic codebook mode (ECB-mode), is to divide the input into sub-blocks and to apply the p.r. permutation on each sub-block separately. This solution suffers from the obvious drawback that every sub-block of output solely depends on a single sub-block of input (and, in particular, the permutation on the complete input is not p.r.). This may leak information about the message being encrypted (see further discussion in Section 6.1).

In this section we consider a generalization of the construction of Section 3 that uses p.r. functions (or permutations) on *small* blocks to construct strong p.r. permutations on *large* blocks. The idea is as follows: apply a pair-wise independent permutation on the entire input, divide the value you get into sub-blocks and apply two rounds of Feistel-permutations (or one round of a p.r. permutation) on each sub-block separately, finally, apply a second pair-wise independent permutation on the entire value you get (see Figure 3 for an illustration).

This solution resembles the electronic codebook mode and is almost as simple. But here, the security we achieve is relative to a random permutation applied on *the entire message* and not on each sub-block separately. As is the case with the electronic codebook mode, the construction is highly suitable for parallel implementation.

For simplicity, we only describe the construction using truly random functions (or a truly random permutation). The analysis of the construction when p.r. functions are used follows easily. In addition, we restrict

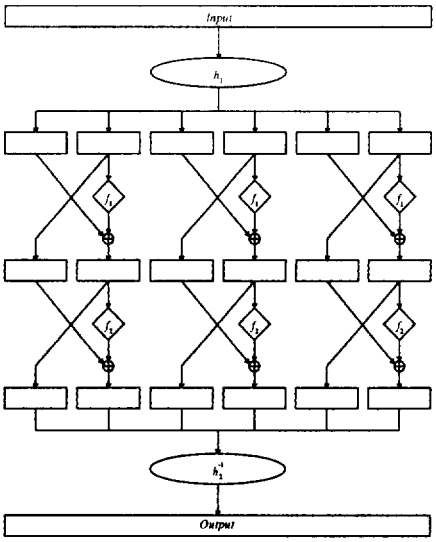

Figure 3: Construction of a strong p.r. permutation on many (six in this case) blocks from a p.r. function on a single block.

our attention to the construction of *strong* p.r. permutations.

Definition 6.1 *For any integers b and s, for any function $g \in F_s$ let $g^{\times b} \in F_{b \cdot s}$ be the function defined by:*

$$g^{\times b}(x_1, x_2, \ldots, x_b) \stackrel{\text{def}}{=} (g(x_1), g(x_2), \ldots, g(x_b))$$

For any $f_1, f_2 \in F_n$ and $h_1, h_2 \in P_{2nb}$, define:

$$S(h_1, f_1, f_2, h_2) \stackrel{\text{def}}{=} h_2^{-1} \circ D_{f_2}^{\times b} \circ D_{f_1}^{\times b} \circ h_1$$

For any $p \in P_{2n}$ and $h_1, h_2 \in P_{2nb}$, define:

$$\hat{S}(h_1, p, h_2) \stackrel{\text{def}}{=} h_2^{-1} \circ p^{\times b} \circ h_1$$

Theorem 6.1 *Let $h_1, h_2 \in P_{2nb}$ be pair-wise independent permutations, let $f_1, f_2 \in F_n$ be random functions and $p \in P_{2n}$ a random permutation. Define $S = S(h_1, f_1, f_2, h_2)$ and $\hat{S} = \hat{S}(h_1, p, h_2)$ (as in Definition 6.1) and let $R \in P_{2nb}$ be a random permutation. Then, for any oracle machine M (not necessarily an efficient one) that makes at most m queries:*

$$\begin{aligned} & |\Pr[M^{S,S^{-1}}(1^{2nb}) = 1] - \Pr[M^{R,R^{-1}}(1^{2nb}) = 1]| \\ \leq\ & \frac{m^2 \cdot b^2}{2^n} + \frac{m^2}{2^{2nb}} \quad \textit{and} \\ & |\Pr[M^{\hat{S},\hat{S}^{-1}}(1^{2nb}) = 1] - \Pr[M^{R,R^{-1}}(1^{2nb}) = 1]| \\ \leq\ & \frac{m^2 \cdot b^2}{2^{2n-1}} \end{aligned}$$

The proof of Theorem 6.1 for S follows the framework described in Section 4 whereas the proof for $\hat{S}$ only slightly deviates from this framework.

Remark 6.1 *The requirements from the distributions of h_1 and h_2 in Theorem 6.1 can be relaxed. This enables us to significantly* decrease the key size *of the p.r. permutations and to increase their efficiency. This issue is discussed in the full version of the paper.*

6.1 Related Work

The construction presented in this section is certainly not the only solution to the problem at hand. We refer in brief to some additional solutions:

As mentioned above, DES modes of operation were suggested as a way of encrypting long messages. However, none of these modes constitutes a construction of a p.r. permutation. For instance, when using the cipher block chaining mode (CBC-mode), the encryptions of two messages with identical prefix will also have an identical prefix. Note that when the encryption of a message M is $f(M)$, for a p.r. permutation f, then the only information that is leaked on M is whether or not M is equal to previously encrypted messages. Bellare et. al. [6] show that the CBC-mode does define a construction of a p.r. function with small output length. They also provide a formal setting for the analysis of the security of p.r. functions with fixed input and output lengths. Bellare et. al. [5] consider the so called *cascade* construction of a p.r. function with small output length. Bellare and Rogaway [7] show how to use the CBC-mode in order to construct a p.r. *permutation* on large inputs (this is the only work we are aware of that explicitly refers to the problem). The work in their construction is comparable to two applications of the CBC-mode (approximately twice the work of our construction, assuming that h_1 and h_2 are relatively efficient). The security of all these constructions is of similar order to the security of our construction.[2] In contrast to our construction, [5, 6, 7] are all sequential in nature.

A different approach that may be attributed in part to Carter and Wegman [33], is to define a length preserving p.r. function $\tilde{F}$ as $G \circ F \circ h$ where, h is a pairwise independent hash function with short output, F is a length preserving p.r. function on short inputs and G a p.r. (bit) generator. It is now possible to use the LR-Construction in order to get a p.r. permutation on large inputs. Anderson and Biham [4] and Lucks [22] show how to directly apply similar ideas into the LR-Construction.

[2] Our construction (as well as the other constructions described in this section) is vulnerable to a birthday-attack on the size of a single block. However, our construction (as well as the other constructions) reduces the problem of foiling birthday-attacks when constructing a p.r. permutation on *many* blocks to the problem of foiling birthday-attacks when constructing a p.r. function (or permutation) on *two* blocks. A solution to the latter is proposed by Aiello and Venkatesan [1].

7 Conclusion and Further Work

The constructions described in Sections 3 & 6 are optimal in their cryptographic work in the sense that the total number of bits on which the cryptographic functions are applied on is exactly the number of bits in the input. Therefore, it seems that in order to achieve the goal of constructing efficient block-ciphers it is sufficient to concentrate on the construction of efficient p.r. functions. The depth of the constructions, on the other hand, is twice the depth of the cryptographic functions. It is an interesting question whether there can be a construction of similar depth. The goal of reducing the depth is even more significant in the case of the $t+2$-round construction in Section 5. A different question is finding a simple construction of k-wise δ-dependent permutations for an *arbitrarily small* δ and an arbitrary k. This question is discussed in [26].

Acknowledgments

We thank Ran Canetti, Oded Goldreich, Kobbi Nissim and Benny Pinkas for many helpful discussions and for their diligent reading of the paper. It is difficult to overestimate Oded's contribution to the presentation of this paper.

References

[1] W. Aiello and R. Venkatesan, Foiling Birthday Attacks in Length-Doubling Transformations, *EUROCRYPT '96*, LNCS, Springer-Verlag, 1996.

[2] N. Alon, L. Babai and A. Itai, A fast and simple randomized parallel algorithm for the maximal independent set problem, *J. Algorithms*, vol. 7(4), 1986, pp. 567-583.

[3] N. Alon, O. Goldreich, J. Hastad and R. Peralta, Simple constructions for almost k-wise independent random variables, *Random Structures and Algorithms*, vol. 3, 1992, pp. 289-304.

[4] R. Anderson and E. Biham, Two practical and provably secure block ciphers: BEAR and LION, *Proc. FSE*, LNCS, vol. 1039, Springer-Verlag, 1996, pp. 113-120.

[5] M. Bellare, R. Canetti and H. Krawczyk, Pseudorandom functions revisited: the cascade construction, *Proc. 37th FOCS*, 1996, pp. 514-523.

[6] M. Bellare, J. Kilian and P. Rogaway, The security of cipher block chaining, *CRYPTO '94*, LNCS, vol. 839, Springer-Verlag, 1994, pp. 341-358.

[7] M. Bellare and P. Rogaway, Block cipher mode of operation for secure, length-preserving encryption, manuscript in preparation.

[8] M. Blum and S. Micali, How to generate cryptographically strong sequence of pseudo-random bits, *SIAM J. Comput.*, vol. 13, 1984, pp. 850-864.

[9] G. Brassard, **Modern cryptology**, LNCS, vol. 325, Springer-Verlag, 1988.

[10] L. Carter and M. Wegman, Universal hash functions, *JCSS*, vol. 18, 1979, pp. 143-154.

[11] B. Chor and O. Goldreich, On the power of two-point based sampling, *J. Complexity*, vol. 5, 1989, pp. 96-106.

[12] S. Even and Y. Mansour, A construction of a cipher from a single pseudorandom permutation, To appear in *J. of Cryptology*. Preliminary version in *ASIACRYPT '91*, LNCS, Springer-Verlag, 1991.

[13] O. Goldreich, **Foundations of cryptography (fragments of a book)**, 1995. Electronic publication: http://www.eccc.uni-trier.de/eccc/info/ECCC-Books/eccc-books.html (Electronic Colloquium on Computational Complexity).

[14] O. Goldreich, S. Goldwasser and S. Micali, How to construct random functions, *J. of the ACM.*, vol. 33, 1986, pp. 792-807.

[15] O. Goldreich, S. Goldwasser and S. Micali, On the cryptographic applications of random functions, *CRYPTO '84*, LNCS, vol. 196, Springer-Verlag, 1985, pp. 276-288.

[16] J. Hastad, R. Impagliazzo, L. A. Levin and M. Luby, Construction of a pseudo-random generator from any one-way function, To appear in *SIAM J. Comput.* Preliminary versions by Impagliazzo et. al. in *21st STOC*, 1989 and Hastad in *22nd STOC*, 1990.

[17] J. Kilian and P. Rogaway, How to protect DES against exhaustive key search, *CRYPTO '96*, 1996, pp. 252-267.

[18] T. Koren, *On the construction of pseudorandom block ciphers*, M.Sc. Thesis (in Hebrew), CS Dept., Technion, Israel, May 1989.

[19] M. Luby, A simple parallel algorithm for the maximal independent set problem, *SIAM J. Comput.*, vol 15(4), 1986, pp. 1036-1053.

[20] M. Luby, **Pseudo-randomness and applications**, Princeton University Press, 1996.

[21] M. Luby and C. Rackoff, How to construct pseudorandom permutations and pseudorandom functions, *SIAM J. Comput.*, vol. 17, 1988, pp. 373-386.

[22] S. Lucks, Faster Luby-Rackoff ciphers, *Proc. FSE*, LNCS, vol. 1039, Springer-Verlag, 1996, pp. 189-203.

[23] U. M. Maurer, A simplified and generalized treatment of Luby-Rackoff pseudorandom permutation generators, *EUROCRYPT '92*, LNCS, Springer-Verlag, 1992, pp. 239-255.

[24] J. Naor and M. Naor, Small-bias probability spaces: efficient constructions and applications, *SIAM J. Comput.*, vol. 22(4), 1993, pp. 838-856.

[25] M. Naor and O. Reingold, Synthesizers and their application to the parallel construction of pseudo-random functions, *Proc. 36th FOCS*, 1995, pp. 170-181.

[26] M. Naor and O. Reingold, On the construction of pseudo-random permutations: Luby-Rackoff revisited, *Theory of Cryptography Library: Record 96-11* at: http://theory.lcs.mit.edu/~tcryptol/homepage.html.

[27] National Bureau of Standards, Data encryption standard, *Federal Information Processing Standard*, U.S. Department of Commerce, FIPS PUB 46, Washington, DC, 1977.

[28] J. Patarin, Pseudorandom permutations based on the DES scheme, *Proc. of EUROCODE '90*, LNCS, Springer-Verlag, 1991, pp. 193-204.

[29] J. Patarin, New results on pseudorandom permutation generators based on the DES scheme, *CRYPTO '91*, LNCS, Springer-Verlag, 1991. pp. 301-312

[30] J. Patarin, Improved security bounds for pseudorandom permutations, To appear in: *4th ACM Conference on Computer and Communications Security*, 1997.

[31] B. Schneier and J. Kelsey, Unbalanced Feistel networks and block cipher design, *Proc. FSE*, LNCS, vol. 1039, Springer-Verlag, 1996, pp. 121-144.

[32] C. P. Schnorr, On the construction of random number generators and random function generators, *EUROCRYPT '88*, LNCS, vol. 330, Springer-Verlag, 1988, pp. 225-232.

[33] M. Wegman and L. Carter, New hash functions and their use in authentication and set equality, *JCSS*, vol. 22, 1981, pp. 265-279.

[34] A. C. Yao, Theory and applications of trapdoor functions, *Proc. 23rd FOCS*, 1982, pp. 80-91.