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

Moni Naor *   Omer Reingold †

Abstract

Luby and Rackoff [27] 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-length using pseudo-random functions with *small* input-length.

*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 and by BSF grant no. 94-00032. 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 permutations, which were introduced by Luby and Rackoff [27], 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) [33].

The advantage of block ciphers (compared with using pseudo-random functions for private-key encryption) 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 pseudo-random permutations in analogy to the different attacks considered in the context of block ciphers:

- Pseudo-random 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 pseudo-random 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.

Pseudo-random permutations are closely related (both in definition and in their construction) to the earlier concept of pseudo-random functions which was defined by Goldreich, Goldwasser and Micali [17]. These are efficiently samplable and computable functions that are indistinguishable from random functions under all (efficient) black-box attacks (see Section 2 for a formal definition). Pseudo-random functions play a major role in private-key cryptography and have many additional applications (for some of these applications, see [11, 18, 26]).

Luby and Rackoff [27] provided a construction of strong pseudo-random permutations, (**LR-Construction**) which was motivated by the structure of DES. The basic building block is the so called Feistel permutation[1] based on a pseudo-random function defined by the key. Their construction consists of four rounds of Feistel permutations (or three rounds, for pseudo-random permutations) each round involves an application of a (different) pseudo-random 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 [17] showed a construction of pseudo-random functions from pseudo-random generators [10, 51]. Thus, the construction of pseudo-random permutations reduces to the construction of pseudo-random generators. Recently a different construction of pseudo-random functions was introduced by Naor and Reingold [32]; this is a parallel construction based on a new primitive called a pseudo-random *synthesizer* that in particular can be constructed from any trapdoor permutation. This implies a parallel construction of pseudo-random permutations. Nevertheless, all known constructions of pseudo-random functions involve non-trivial (though of course polynomial-time) computation, so it makes sense to attempt to minimize the number of invocations of pseudo-random functions.

[1]A Feistel permutation for a function $f : \{0,1\}^n \mapsto \{0,1\}^n$ is a permutation on $\{0,1\}^{2n}$ defined by $\mathbf{D}_f(L,R) \stackrel{\text{def}}{=} (R, L \oplus f(R))$, where $|L| = |R| = n$. Each of the 16 rounds of DES involves a Feistel permutation of a function determined by the 56 key bits.

Alongside cryptographic pseudo-randomness the last two decades saw the development of the notion of limited independence in various setting and formulations [3, 4, 13, 14, 25, 31, 50]. 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) pseudo-random 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. Simplify the proof of security of the construction (especially in the case of strong pseudo-random permutations) and provide a framework for proving the security of similar constructions.

2. 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 framework.

3. Achieve an improvement in the computational complexity of the pseudo-random permutations – two applications of a pseudo-random function on n bits suffice for computing the value of a pseudo-random permutation on $2n$ bits at a given point (vs. four applications in the original LR-Construction). This implies that the reduction is "optimal".

As discussed in Section 5.2, 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 following is a brief and informal description of the paper's main results and organization:

Section 2 Reviews the notation and the definitions regarding pseudo-randomness and k-wise independence.

Section 3 Presents the main construction and proves its security: pair-wise independent permutations can replace the first and fourth rounds of the LR-Construction (see Figure 1.b for an illustration).

Section 4 Highlights the high-level structure of the proof of security which provides a framework that enables us to relax and generalize the main construction.

Section 5 Shows how the main construction can be relaxed by:

- **5.1** Using a single pseudo-random function (instead of two) and
- **5.2** Using weaker and more efficient permutations (or functions) instead of the pair-wise independent permutations.

Section 6 Provides a simple generalization of the main construction: using t rounds of (generalized) Feistel permutations (instead of two) the success probability of the distinguisher is reduced from approximately $\frac{m^2}{2^{\ell/2}}$ to approximately $\frac{t}{2} \cdot \frac{m^2}{2^{(1-1/t)\ell}}$, where the permutation is on ℓ bits and the distinguisher makes at most m queries (see Figure 3 for an illustration).

Section 7 Provides a second generalization of the main construction. Instead of applying Feistel permutations on the entire outputs of the first and second rounds, Feistel permutations can be separately applied on each one of their sub-blocks. This is a construction of a strong pseudo-random permutation on *many* blocks using pseudo-random functions on a *single* block (see Figure 4 for an illustration).

Section 8 Analyzes the different constructions of the paper as constructions of k-wise δ-dependent permutations.

Section 9 Suggests directions for further research.

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 [29] 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 indistinguishable under non-adaptive attacks and can be distinguished using a very simple adaptive attack.[2] A different approach toward the proof was described by Patarin [35] (this is the only published proof, we are aware of, for the LR-Construction of *strong* pseudo-random permutations; another proof was given by Koren [23]).

Other papers consider the security of possible variants of the construction. A significant portion of this research deals with the construction of pseudo-random permutations and strong pseudo-random permutations from a *single* pseudo-random function. This line of work is described in Section 5.1.

Lucks [28] shows that a hash function can replace the pseudo-random function in the first round of the three-round LR-Construction. His proof is based on [29] 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 pseudo-random permutations.

Somewhat different questions were considered by Even and Mansour [15] and by Kilian and Rogaway [22]. Loosely speaking, the former construct several pseudo-random permutations from a single one, while the latter show how to make exhaustive key-search attacks more difficult. The construction itself amounts, in both cases, to XORing the input of the pseudo-random permutation with a random key and XORing the output of the permutation with a second random key.

The background and related work concerning other relevant issues are discussed in the appropriate sections: Definitions and constructions of efficient hash functions in Section 5.2, reducing the distinguishing probability in Section 6 and the construction of pseudo-random permutations (or functions) with large input-length from pseudo-random permutations (or functions) with small input-length in Section 7.

[2] An even more striking example is obtained by comparing a random permutation P that satisfies $P(P(0)) = 0$ with a truly random permutation.

2 Preliminaries

In this section, the concepts of pseudo-random functions and pseudo-random permutations are briefly reviewed. A more thorough and formal treatment can be found in [16, 26].

2.1 Notation

- 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) [3] *For any function $f \in F_n$, let $\mathbf{D}_f \in P_{2n}$ be the permutation defined by $\mathbf{D}_f(L, R) \stackrel{\text{def}}{=} (R, L \oplus f(R))$, where $|L| = |R| = n$.*

Notice that Feistel permutations are as easy to invert as they are to compute (since the inverse permutation satisfies $\mathbf{D}_f^{-1}(L, R) = (R \oplus f(L), L)$; that is, $\mathbf{D}_f^{-1}(L, R) \equiv \rho \circ \mathbf{D}_f \circ \rho$ for $\rho(L, R) \stackrel{\text{def}}{=} (R, L)$). Therefore, the LR-Construction (and its different variants which are introduced in Sections 6 & 7) 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, pseudo-random distributions cannot be efficiently distinguished from the truly random distributions (usually, random here means uniform). However, the pseudo-random distributions have substantially smaller entropy than the truly random distributions and are efficiently samplable.

2.2.1 Overview of Pseudo-Random Primitives

In the case of **pseudo-random (bit) generators**, which were introduced by Blum and Micali and Yao [10, 51], the pseudo-random 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 [21] showed how to construct a pseudo-random 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).

Pseudo-random function ensembles (PFE), which were introduced by Goldreich, Goldwasser and Micali [17], 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 pseudo-random generators.

[3] $\mathbf{D}$ stands for DES-like, another common term for these permutations.

Luby and Rackoff [27] define **pseudo-random 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 pseudo-random permutation generators*. Here the distinguisher can also access the inverse permutation at points of its choice. Following [16] we use the term **strong pseudo-random 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 [33], namely, the compositions of several Feistel-permutations. Their design of the PPE (resp. SPPE) is $\mathbf{D}_{f_3} \circ \mathbf{D}_{f_2} \circ \mathbf{D}_{f_1}$ (resp. $\mathbf{D}_{f_4} \circ \mathbf{D}_{f_3} \circ \mathbf{D}_{f_2} \circ \mathbf{D}_{f_1}$) where all f_is are independent pseudo-random functions and $\mathbf{D}_{f_i}$ as in Definition 2.1 (see Figure 1.a for an illustration).

2.2.2 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 ensemble (or a permutation ensemble), $H = \{H_n\}_{n \in \mathbb{N}}$, is *efficiently computable* if the distribution H_n can be sampled efficiently and the functions in H_n can be computed efficiently. I.e., there exist probabilistic polynomial-time Turing-machines, I and V, and a mapping from strings to functions, ϕ, such that $\phi(I(1^n))$ and H_n are identically distributed and $V(i, x) = (\phi(i))(x)$ (so, in fact, $H_n \equiv V(I(1^n), \cdot)$).

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. An oracle machine has an interpretation both under the uniform complexity model and under the non-uniform model. In the former it is interpreted as a Turing-machine with a special oracle-tape (in this case efficient means probabilistic polynomial-time) and in the latter as a circuit-family with special oracle-gates (in this case efficient means polynomial-size). The discussion of this paper is independent of the chosen interpretation.

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*

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

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

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

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

Definition 2.3 (ϵ-distinguish) *We say that M ϵ-distinguishes between H and $\tilde{H}$ (resp. $\langle H, H^{-1}\rangle$ and $\langle \tilde{H}, \tilde{H}^{-1}\rangle$) for $\epsilon = \epsilon(n)$ 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 2.4 (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 2.5 (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* pseudo-random function ensemble *if for every efficient oracle-machine M, the advantage M has in distinguishing between H and R is negligible.*

Definition 2.6 (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* pseudo-random permutation ensemble *if for every efficient oracle-machine M, the advantage M has in distinguishing between H and R is negligible.*

Definition 2.7 (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 pseudo-random 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 pseudo-random function" as an abbreviation for "f is distributed according to a pseudo-random function ensemble" and similarly for "f is a pseudo-random permutation" and "f is a strong pseudo-random permutation".*

2.3 k-Wise Independent Functions and Permutations

The notions of k-wise independent functions and k-wise "almost" independent functions [3, 4, 13, 14, 25, 31, 50] (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.

We briefly review the definitions of k-wise independence (and k-wise δ-dependence). The definitions of pair-wise independence (and pair-wise δ-dependence) can be derived by taking $k = 2$.

Definition 2.8 *Let D_1 and D_2 be two distributions defined over Ω, the variation distance between D_1 and D_2 is*

$$\|D_1 - D_2\| = \frac{1}{2}\sum_{\omega\in\Omega} |D_1(\omega) - D_2(\omega)|$$

Definition 2.9 *Let A and B be two sets, $0 \leq \delta \leq 1$, k an integer ($2 \leq k \leq |A|$) and F a distribution of $A \mapsto B$ functions.*

Let $x_1, x_2, \ldots, x_k$ be k different members of A, consider the following two distributions:

1. *$\langle f(x_1), f(x_2), \ldots, f(x_k) \rangle$ where f is distributed according to F.*

2. *The uniform distribution over B^k.*

F is k-wise independent if for all $x_1, x_2, \ldots, x_k$ the two distributions are identical. F is k-wise δ-dependent if for all $x_1, x_2, \ldots, x_k$ the two distributions are of variation distance at most δ.

These definitions are naturally extended to permutations:

Definition 2.10 *Let A be a set, $0 \leq \delta \leq 1$, k an integer ($2 \leq k \leq |A|$) and F a distribution of permutations over A.*

Let $x_1, x_2, \ldots, x_k$ be k different members of A, consider the following two distributions:

1. *$\langle f(x_1), f(x_2), \ldots, f(x_k) \rangle$ where f is distributed according to F.*

2. *The uniform distribution over sequences of k* different *elements of A.*

F is k-wise independent if for all $x_1, x_2, \ldots, x_k$ the two distributions are identical. F is k-wise δ-dependent if for all $x_1, x_2, \ldots, x_k$ the two distributions are of variation distance at most δ.

The connection of this paper to k-wise independence is bidirectional as described in the following two paragraphs.

As shown in Section 3, pair-wise independent permutations can replace the first and fourth rounds of the LR-Construction. 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 Section 5.2, it is shown that we can use even more efficient functions and permutations in our construction. In particular, we consider the concept of ϵ-AXU$_2$ functions.

In contrast with the case of pair-wise independent permutations, we are not aware of any "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 pseudo-random functions in the original LR-Construction yields a k-wise δ-dependent permutation on $2n$ bits (for $\delta = O(k^2/2^n + \delta')$). In Section 8 we analyze the different constructions of this paper as constructions of k-wise δ-dependent permutations.

3 Construction of PPE and SPPE

3.1 Intuition

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, $E = \mathbf{D}_{f_2} \circ \mathbf{D}_{f_1}$, where $f_1, f_2 \in F_n$ are two independently chosen pseudo-random 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. The intuition is as follows: Note that it is

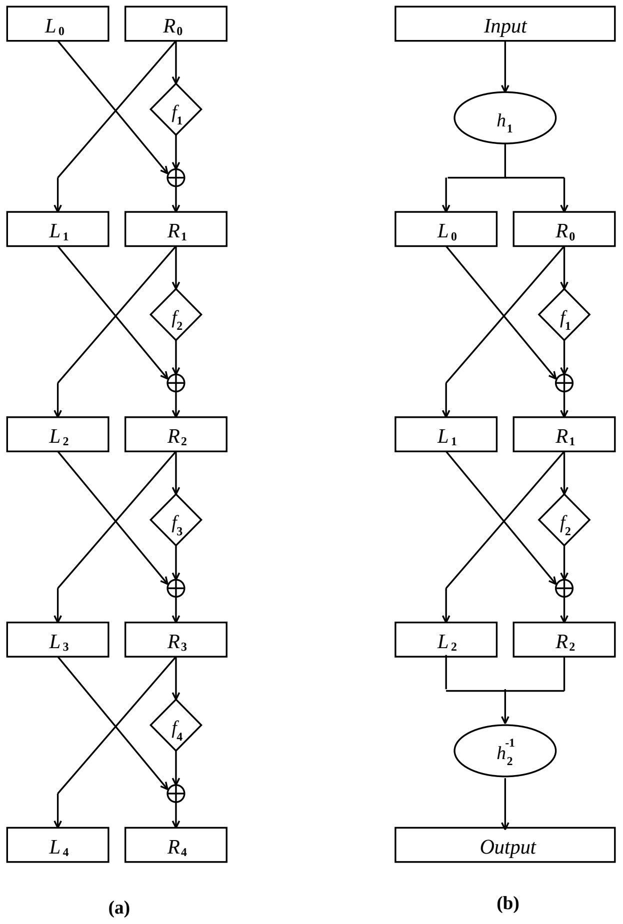

Figure 1: Constructions of SPPE: (a) The original LR-Construction (b) The revised Construction. In (a) and (b): $\forall i \geq 1$, $L_i = R_{i-1}$ and $R_i = L_{i-1} \oplus f_i(R_{i-1})$. In (b): $\langle L_0, R_0 \rangle = h_1(Input)$ and $Output = h_2^{-1}(\langle L_2, R_2 \rangle)$.

enough to prove the pseudo-randomness of E when f_1 and f_2 are *truly random functions* (instead of pseudo-random). Let $(L_i^0, R_i^0) = x_i$ and $(L_i^2, R_i^2) = E(x_i)$, by the definition of E we get that $L_i^2 = L_i^0 \oplus f_1(R_i^0)$ and $R_i^2 = R_i^0 \oplus f_2(L_i^2)$. Since the sequence $\{x_i\}_{i=1}^m$ is uniformly distributed, we have that with good probability (better than $(1 - \frac{m^2}{2^{n+1}})$) $R_i^0 \neq R_j^0$ for all $i \neq j$. Conditioned on this event, the sequence $\{L_i^2\}_{i=1}^m$ is uniformly distributed and independent of the sequence $\{x_i\}_{i=1}^m$ (since f_1 is random). We now have that with good probability $L_i^2 \neq L_j^2$ for all $i \neq j$. Conditioned on this event, the sequence $\{R_i^2\}_{i=1}^m$ is uniformly distributed and independent of both $\{L_i^2\}_{i=1}^m$ and $\{x_i\}_{i=1}^m$. Notice that this argument still works if the sequence $\{x_i\}_{i=1}^m$ is only pair-wise independent.

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. The reason is that for any values L_1, L_2 and R such that $L_1 \neq L_2$ we have that $E(L_1, R)_{|_L} \oplus E(L_2, R)_{|_L} = L_1 \oplus L_2$. In contrast, for a truly random permutation, the probability of this event is 2^{-n}. This is the reason that the LR-Construction includes three or four rounds.

If we think of the second and third rounds of the LR-Construction as the permutation E, then the discussion above implies that the role of the first and fourth rounds is to prevent the distinguisher from directly choosing the inputs of E and E^{-1}. We show that this goal can also be achieved with "combinatorial" constructions (e.g., pair-wise independent permutations), rather than "cryptographic" (i.e., pseudo-random functions). In particular, the LR-Construction remains secure when the first and fourth rounds are replaced with pair-wise independent permutations (see Figure 1 for an illustration).

3.2 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*

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

and

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

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 pseudo-random functions; h_1, h_2, f_1 and f_2 are independently chosen. Then, $W = W(h_1, f_1, f_2)$ is a pseudo-random permutation and $S = S(h_1, f_1, f_2, h_2)$ is a strong pseudo-random permutation (W and S as in Definition 3.1).*

Furthermore, assume that no efficient oracle-machine that makes at most $m = m(n)$ queries, ϵ-distinguishes between the pseudo-random functions and random functions for $\epsilon = \epsilon(n)$ (see Definition 2.3). Then, no efficient oracle-machine that makes at most m queries to W (resp. S and S^{-1}) ϵ'-distinguishes W (resp. S) from a random permutation for $\epsilon' = 2\epsilon + \frac{m^2}{2^n} + \frac{m^2}{2^{2n}}$

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 Section 5. The main points are the following:*

1. *A single pseudo-random 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 $x \neq y$:*

$$\Pr[h_1(x)_{|_R} = h_1(y)_{|_R}] \leq 2^{-n} \text{ and } \Pr[h_2(x)_{|_L} = h_2(y)_{|_L}] \leq 2^{-n}$$

3.3 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 pseudo-random when f_1 and f_2 are truly random (instead of pseudo-random).

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:*

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

Theorem 3.1 follows easily from Theorem 3.2 (see a proof-sketch in the sequel). In order to prove Theorem 3.2, we introduce additional notation.

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 i^{th} 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 i^{th} 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 i^{th} 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 i^{th} 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 σ be a sequence $\{\langle x_1, y_1 \rangle, ..., \langle x_m, y_m \rangle\}$, where for $1 \leq i \leq m$ we have that $\langle x_i, y_i \rangle \in I_{2n} \times I_{2n}$. 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 the ith query of M answers as follows:

1. If M's query is $(+, x)$ and for some $1 \leq j < i$ the j^{th} query-answer pair is $\langle x, y \rangle$, then $\tilde{R}$'s answer is y (for an arbitrary such query-answer pair, $\langle x, y \rangle$).

2. If M's query is $(-, y)$ and for some $1 \leq j < i$ the j^{th} query-answer pair is $\langle x, y \rangle$, then $\tilde{R}$'s answer is x (for an arbitrary such query-answer pair, $\langle x, y \rangle$).

3. If neither 1 nor 2 holds, then $\tilde{R}$'s answer is a uniformly chosen $2n$-bit string.

It is possible that $\tilde{R}$ provides answers that are not consistent with *any* permutation:

Definition 3.3 *Let $\sigma = \{\langle x_1, y_1\rangle, ..., \langle x_m, y_m\rangle\}$ be any possible M-transcript. σ is* inconsistent *if for some $1 \leq j < i \leq m$ the corresponding query-answer pairs satisfy $x_i = x_j$ and $y_i \neq y_j$ or $y_i = y_j$ and $x_i \neq x_j$. Otherwise, σ is* 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 3.4 *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

$$\left|\Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1] - \Pr_R[C_M(T_R) = 1]\right| \leq \frac{m^2}{2^{2n+1}}$$

Proof: For any possible and consistent M-transcript σ we have that

$$\Pr_R[T_R = \sigma] = \frac{2^{2n}!}{(2^{2n} - m)!} = \Pr_{\tilde{R}}[T_{\tilde{R}} = \sigma \mid T_{\tilde{R}} \textit{ is consistent}].$$

Therefore, the distribution of $T_{\tilde{R}}$ *conditioned* on $T_{\tilde{R}}$ being consistent is exactly the distribution of T_R. Furthermore, the probability that $T_{\tilde{R}}$ is inconsistent is small: $T_{\tilde{R}}$ is inconsistent if for some $1 \leq j < i \leq m$ the corresponding query-answer pairs satisfy $x_i = x_j$ and $y_i \neq y_j$ or $y_i = y_j$ and $x_i \neq x_j$. For a given i and j this event happens with probability at most 2^{-2n}. Hence,

$$\Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is inconsistent}] \leq \binom{m}{2} \cdot 2^{-2n} < \frac{m^2}{2^{2n+1}}$$

The proposition follows:

$$\begin{aligned}
& \left|\Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1] - \Pr_R[C_M(T_R) = 1]\right| \\
\leq \; & \left|\Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1 \mid T_{\tilde{R}} \textit{ is consistent}] - \Pr_R[C_M(T_R) = 1]\right| \cdot \Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is consistent}] \\
& + \left|\Pr_{\tilde{R}}[C_M(T_{\tilde{R}}) = 1 \mid T_{\tilde{R}} \textit{ is inconsistent}] - \Pr_R[C_M(T_R) = 1]\right| \cdot \Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is inconsistent}] \\
\leq \; & \Pr_{\tilde{R}}[T_{\tilde{R}} \textit{ is inconsistent}] \\
< \; & \frac{m^2}{2^{2n+1}}
\end{aligned}$$

□

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 3.5) and bound its probability (Proposition 3.4).

Convention 3.1 *For any possible M-transcript $\sigma = \{\langle x_1, y_1\rangle, ..., \langle x_m, y_m\rangle\}$ we can assume hereafter that if σ is consistent then 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 3.5 *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:*

$$\exists 1 \leq i < j \leq m \text{ such that } h_1(x_i)_{|R} = h_1(x_j)_{|R} \text{ or } h_2(y_i)_{|L} = h_2(y_j)_{|L}$$

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:*

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

Proof: By definition, $\sigma \in \mathrm{BAD}(h_1, h_2)$ if there exist $1 \leq i < j \leq m$ such that either $h_1(x_i)_{|R} = h_1(x_j)_{|R}$ or $h_2(y_i)_{|L} = h_2(y_j)_{|L}$. For any given i and j both $\Pr_{h_1}[h_1(x_i)_{|R} = h_1(x_j)_{|R}]$ and $\Pr_{h_2}[h_2(y_i)_{|L} = h_2(y_j)_{|L}]$ are smaller than 2^{-n} (since h_1 and h_2 are pair-wise independent). Therefore,

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

□

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 for all $1 \leq i \leq m$,

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

Therefore, $T_{\tilde{R}} = \sigma$ iff for all $1 \leq i \leq m$, the i^{th} 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 i^{th} answer is x_i. Assume that $\tilde{R}$ answered "correctly" (i.e., y_i or x_i as above) for each one of the first $i-1$ queries. Then by Convention 3.1 and the definition of $\tilde{R}$ its i^{th} 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 $M(\sigma)=1$.

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

We already showed in the proof of Proposition 3.3 that the value of Expression (3) is smaller than $\frac{m^2}{2^{2n+1}}$, by Lemma 3.5 we get that the value of Expression (1) is 0. Therefore, it remains to bound the value of Expression (2): Assume without loss of generality that

$$\begin{aligned}
& \sum_{\sigma\in\Gamma}\Pr_S[T_S=\sigma\mid\sigma\in \mathrm{BAD}(h_1,h_2)]\cdot\Pr_{h_1,h_2}[\sigma\in \mathrm{BAD}(h_1,h_2)] \\
\leq\ & \sum_{\sigma\in\Gamma}\Pr_{\tilde{R}}[T_{\tilde{R}}=\sigma]\cdot\Pr_{h_1,h_2}[\sigma\in \mathrm{BAD}(h_1,h_2)]
\end{aligned}$$

then using Proposition 3.4 we get that

$$\begin{aligned}
& \left|\sum_{\sigma\in\Gamma}\left(\Pr_S[T_S=\sigma\mid\sigma\in \mathrm{BAD}(h_1,h_2)]-\Pr_{\tilde{R}}[T_{\tilde{R}}=\sigma]\right)\cdot\Pr_{h_1,h_2}[\sigma\in \mathrm{BAD}(h_1,h_2)]\right| \\
\leq\ & \sum_{\sigma\in\Gamma}\Pr_{\tilde{R}}[T_{\tilde{R}}=\sigma]\cdot\Pr_{h_1,h_2}[\sigma\in \mathrm{BAD}(h_1,h_2)] \\
\leq\ & \max_{\sigma\in\Gamma}\Pr_{h_1,h_2}[\sigma\in \mathrm{BAD}(h_1,h_2)] \\
<\ & \frac{m^2}{2^n}
\end{aligned}$$

Thus, we can conclude that:

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

Using Proposition 3.3 we complete the proof:

$$\begin{aligned}
& \left|\Pr_S[M^{S,S^{-1}}(1^{2n})=1]-\Pr_R[M^{R,R^{-1}}(1^{2n})=1]\right| \\
=\ & \left|\Pr_S[C_M(T_S)=1]-\Pr_R[C_M(T_R)=1]\right| \\
\leq\ & \left|\Pr_S[C_M(T_S)=1]-\Pr_{\tilde{R}}[C_M(T_{\tilde{R}})=1]\right| + \left|\Pr_{\tilde{R}}[C_M(T_{\tilde{R}})=1]-\Pr_R[C_M(T_R)=1]\right| \\
<\ & \frac{m^2}{2^n}+\frac{m^2}{2^{2n}}
\end{aligned}$$

□

Given Theorem 3.2, the proof of Theorem 3.1 is essentially the same as the corresponding proof of the original LR-Construction (the proof of Theorem 1 of [27], given their main Lemma). The proof-idea is the following: Define three distributions:

- $S_1 = S(h_1, f_1, f_2, h_2)$, where $h_1, h_2 \in P_{2n}$ are pair-wise independent and $f_1, f_2 \in F_n$ are pseudo-random functions.
- $S_2 = S(h_1, g_1, f_2, h_2)$, where $h_1, h_2 \in P_{2n}$ are pair-wise independent, $f_2 \in F_n$ a pseudo-random function and $g_1 \in F_n$ a random function.
- $S_3 = S(h_1, g_1, g_2, h_2)$, where $h_1, h_2 \in P_{2n}$ are pair-wise independent and $g_1, g_2 \in F_n$ are random functions.

It is enough to show that for every oracle machine, for all but finite number of n's:

1. $\left|\Pr[M^{S_1,S_1^{-1}}(1^{2n}) = 1] - \Pr[M^{S_2,S_2^{-1}}(1^{2n}) = 1]\right| \leq \epsilon(n)$
2. $\left|\Pr[M^{S_2,S_2^{-1}}(1^{2n}) = 1] - \Pr[M^{S_3,S_3^{-1}}(1^{2n}) = 1]\right| \leq \epsilon(n)$

If (1) or (2) do not hold, then we can construct an efficient oracle-machine M' that ϵ-distinguishes the pseudo-random functions from the random functions in contradiction to the assumption. Assume for example that for infinitely many n's:

$$\left|\Pr[M^{S_1,S_1^{-1}}(1^{2n}) = 1] - \Pr[M^{S_2,S_2^{-1}}(1^{2n}) = 1]\right| > \epsilon(n)$$

The oracle-machine M' on input 1^n and with access to a function $O \in F_n$ first samples pair-wise independent permutations, $h_1, h_2 \in P_{2n}$, and a pseudo-random function $f_2 \in F_n$. M' then invokes M with input 1^{2n} and answers its queries with the values of S and S^{-1}, for $S = S(h_1, O, f_2, h_2)$. When M halts so does M' and it outputs whatever was the output of M. Notice that if O is a pseudo-random function then the distribution of S is S_1 whereas if O is a truly random function then the distribution of S is S_2. This is the reason that M' distinguishes a pseudo-random function from a random one with advantage greater than $\epsilon(n)$. Similar hybrid-arguments apply to all other constructions of this paper.

4 The Framework

As we shall see in Sections 5–7, the construction of Section 3 can be relaxed and generalized in several ways. The different pseudo-random 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 pseudo-random permutation S on ℓ bits which is the composition of three permutations: $S \equiv h_2^{-1} \circ E \circ h_1$. (see Figure 2 for an illustration). In general, h_1 and h_2^{-1} are "lightweight" and E is where most of the work is done. E is constructed from pseudo-random 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 $E \equiv \mathbf{D}_{f_2} \circ \mathbf{D}_{f_1}$ 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 pseudo-random 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:

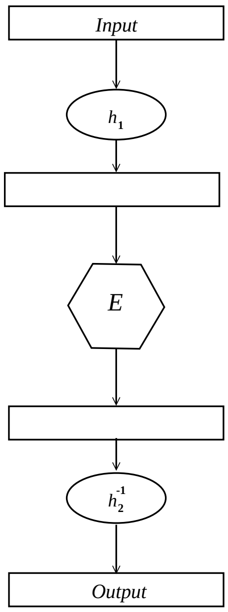

Figure 2: The high-level structure of the different constructions of SPPE.

Definition 4.1 *A sequence,* $\{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$*, is* E-Good *if* $\Pr_E[\forall i,\ y_i = E(x_i)] = 2^{-\ell \cdot m}$.

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:

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

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 (saying that the distance between T_R and $T_{\tilde{R}}$ is bounded by the probability that $T_{\tilde{R}}$ is inconsistent and that this probability is bounded by $\frac{m^2}{2^{\ell+1}}$) 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 3.5 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 3.5 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

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

and this substitutes Proposition 3.4. This is the only place in the proof where we use the definition of ϵ and the definition of the distributions of h_1 and h_2. As demonstrated in Sections 5.2 & 7.1, there is actually a tradeoff between reducing the requirements from h_1 and h_2 and having a somewhat larger value of ϵ. 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:*

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

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 pseudo-random 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 7).

5 Relaxing the Construction

5.1 PPE and SPPE with a Single Pseudo-Random Function

Since Luby and Rackoff introduced their construction a considerable amount of research [34, 35, 36, 37, 39, 43, 44, 45, 47, 52] was focused on the following question: Can we obtain a similar construction of PPE or SPPE such that every permutation will be constructed from a *single* pseudo-random function?

Apparently, this line of research originated in the work of Schnorr [47]. Schnorr considered the LR-Construction, where the functions used are truly random, as a pseudo-random generator that is secure if not too many bits are accessible. The security of Schnorr's generator does not depend on any unproven assumption. This notion of local-randomness is further treated in [29, 30]. Since the key of a random function is huge it makes sense to minimize the number of functions and, indeed, Schnorr suggested $\mathbf{D}_f \circ \mathbf{D}_f \circ \mathbf{D}_f$ as pseudo-random (the suggested permutation was later shown to be distinguishable from random [43]).

Following is an informal description of some of these results. Let $f \in F_n$ be a random function, then:

- For all $i, j, k \geq 1$ the permutation $\mathbf{D}_{f^i} \circ \mathbf{D}_{f^j} \circ \mathbf{D}_{f^k}$ is not pseudo-random [52].
- For all $i, j, k, \ell \geq 1$ the permutation $\mathbf{D}_{f^i} \circ \mathbf{D}_{f^j} \circ \mathbf{D}_{f^k} \circ \mathbf{D}_{f^\ell}$ is not strongly pseudo-random [44].
- $\mathbf{D}_{f^2} \circ \mathbf{D}_f \circ \mathbf{D}_f \circ \mathbf{D}_f$ is pseudo-random [39].
- $\mathbf{D}_f \circ \mathbf{D}_I \circ \mathbf{D}_{f^2} \circ \mathbf{D}_f \circ \mathbf{D}_I \circ \mathbf{D}_{f^2}$ is strongly pseudo-random, where $I \in F_n$ is the identity function [45].

- $\mathbf{D}_{f\circ\xi\circ f} \circ \mathbf{D}_f \circ \mathbf{D}_f$ is pseudo-random and $\mathbf{D}_{f\circ\xi\circ f} \circ \mathbf{D}_f \circ \mathbf{D}_f \circ \mathbf{D}_f$ is strongly pseudo-random, where ξ is, for example, a rotation of one bit [37].

A critique which has been voiced often is that using only one pseudo-random function does not seem too significant: A pseudo-random function on $n+2$ bits can replace 4 pseudo-random functions on n bits or, alternatively, a small key can be used to pseudo-randomly generate a larger key. It should also be noticed that the new constructions require additional invocations of the pseudo-random functions which imply an increase in the computation time. Furthermore, these results involve detailed and non-trivial proofs (to a point, where some papers claim to find inaccuracies in others).

The adjustment of the LR-Construction we suggest in Section 3 can easily be converted into a construction of PPE and SPPE from a single pseudo-random function. Simply replace both (pseudo-random) functions, f_1 and f_2, with a single (pseudo-random) function f. This solution does not suffer from the drawbacks of the previous ones. The construction and the proof remain as simple as before and the pseudo-random function is only invoked twice at each computation of the permutation. The additional key-length for the pair-wise independent functions (h_1 and h_2) is not substantial (especially compared to the length of a truly random function). Consider, for example, the construction of SPPE when we use a truly random function f:

Theorem 5.1 *Let $h_1, h_2 \in P_{2n}$ be pair-wise independent permutations and let $f \in F_n$ be a random function. Define $S = S(h_1, f, f, h_2)$ (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:*

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

The proof follows the framework described in Section 4. The set $\mathrm{BAD}(h_1, h_2)$ (Definition 3.5) is replaced with the set $\mathrm{BAD}_1(h_1, h_2)$ defined 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 there exist $1 \leq i < j \leq m$ such that either $h_1(x_i)_{|_R} = h_1(x_j)_{|_R}$ or $h_2(y_i)_{|_L} = h_2(y_j)_{|_L}$ (as before),* or *there exist $1 \leq i, j \leq m$ such that $h_1(x_i)_{|_R} = h_2(y_j)_{|_L}$.*

In order to apply Theorem 4.2, it is enough to note that by this definition we get that for any possible and consistent M-transcripts, σ, both $\Pr_S[T_S = \sigma \mid \sigma \notin \mathrm{BAD}_1(h_1, h_2)] = 2^{-2nm}$ (hence, it is a proper definition according to the framework) and $\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}_1(h_1, h_2)] < \frac{2m^2}{2^n}$.

5.2 Relaxing the Pair-Wise Independence Requirement

One might interpret the construction of Section 3 in the following way: given the task of constructing *efficient* pseudo-random permutations it is enough to concentrate on the efficient construction of pseudo-random *functions*. The assumption that supports such a claim is that the computation of pseudo-random functions is much more expensive than the computation of pair-wise independent permutations. Therefore, computing the value of the pseudo-random permutation (that is constructed in Section 3) on any input of $2n$ bits is essentially equivalent to two invocations of a pseudo-random function with n-bit inputs. In this section we show that we can use even weaker permutations instead of the pair-wise independent ones – resulting in an even more efficient construction of pseudo-random permutations.

As mentioned in Section 4, the only place in Section 3 we use the fact that h_1 and h_2 are pair-wise independent permutations is in the proof of Proposition 3.4. In fact, the exact requirement on h_1 and h_2 we use is that for every $x \neq y$:

$$\Pr_{h_1}[h_1(x)_{|R} = h_1(y)_{|R}] \leq 2^{-n} \textit{ and } \Pr_{h_2}[h_2(x)_{|L} = h_2(y)_{|L}] \leq 2^{-n}.$$

Furthermore, we can replace 2^{-n} with any $\epsilon \geq 2^{-n}$ and still get a construction of pseudo-random permutations (with somewhat larger distinguishing probability). Consider, for example, the revised statement of Theorem 3.2:

Theorem 5.2 *Let H^1 and H^2 be distributions of permutations in P_{2n} such that for every pair of $2n$-bit strings $x \neq y$:*

$$\Pr_{h_1 \in H^1}[h_1(x)_{|R} = h_1(y)_{|R}] \leq \epsilon \textit{ and } \Pr_{h_2 \in H^2}[h_2(x)_{|L} = h_2(y)_{|L}] \leq \epsilon$$

Let h_1 be distributed according to H^1, h_2 distributed according to H^2 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:

$$\left|\Pr[M^{S,S^{-1}}(1^{2n}) = 1] - \Pr[M^{R,R^{-1}}(1^{2n}) = 1]\right| < m^2 \cdot \epsilon + \frac{m^2}{2^{2n}}$$

The proof follows the framework described in Section 4. This time the definition of $\mathrm{BAD}(h_1, h_2)$ stays unchanged and, in order to apply Theorem 4.2, we only need to note that for any possible and consistent M-transcript σ, $\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}(h_1, h_2)] < m^2 \cdot \epsilon$.

The conditions on H^1 and H^2 in Theorem 5.2 are somewhat nonstandard (since the requirements are on half the bits of the output). Nevertheless, these conditions are satisfied by more traditional requirements on function-families. In particular, one can use the concept of ϵ-AXU_2 functions:

Definition 5.1 *A distribution on $I_n \mapsto I_n$ functions (or permutations), H, is ϵ-AXU_2 if for every $x \neq y$ and every z ($x, y, z \in I_n$),*

$$\Pr_{h \in H}[h(x) \oplus h(y) = z] \leq \epsilon$$

This concept was originally defined by Carter and Wegman [13]; We use the terminology of Rogaway [41].

It is easy to verify that the conditions on H^1 and H^2 in Theorem 5.2 are satisfied if both H^1 and H^2 are $((2^n - 1)^{-1} \cdot \epsilon)$-$\mathrm{AXU}_2$. Such a distribution of permutations over I_{2n}, for $\epsilon = (2^n + 1)^{-1}$, is $h_a(x) \stackrel{\text{def}}{=} a \cdot x$ where a is uniform in $I_{2n} \setminus \{0\}$ and the multiplication is in $GF(2^{2n})$.

Another way to construct H^1 and H^2 is by *using Feistel permutations* with ϵ-AXU_2 *functions.* Let H be a distribution of ϵ-AXU_2 functions on n bits strings then, we can define H^1 to be $\{\mathbf{D}_h\}_{h \in H}$ and H^2 to be $\{\mathbf{D}_h^{-1}\}_{h \in H}$. The reason is that for every two *different* $2n$-bit strings $x = (L^1, R^1)$ and $y = (L^2, R^2)$ and every function $h \in F_n$ we have by definition that:

$$\mathbf{D}_h(x)_{|R} = \mathbf{D}_h(y)_{|R} \Longleftrightarrow h(R^1) \oplus h(R^2) = L^1 \oplus L^2$$

If $R^1 = R^2$ then $L^1 \neq L^2$ and therefore $\mathbf{D}_h(x)_{|R} \neq \mathbf{D}_h(y)_{|R}$ otherwise, by the definition of ϵ-AXU_2 functions:

$$\Pr_{h \in H}[\mathbf{D}_h(x)_{|R} = \mathbf{D}_h(y)_{|R}] = \Pr_{h \in H}[h(R^1) \oplus h(R^2) = L^1 \oplus L^2] \leq \epsilon$$

Thus, H^1 satisfies its requirement and similarly for H^2.

By using Feistel permutations to construct H^1 and H^2 we get the original LR-Construction as a special case (since a random function is in particular 2^{-n}-AXU$_2$). Thus, the proof of security in Section 3 also holds for the original LR-Construction. The idea of using ϵ-AXU$_2$ functions instead of pseudo-random functions for the first round of the LR-Construction was previously suggested by Lucks [28].

Another advantage of this approach is that it allows us to use many efficient constructions of function families. An example of efficient 2^{-n}-AXU$_2$ functions are Vazirani's "shift"-family [49]. A key of such a function is a uniformly chosen string $a \in I_{2n-1}$ and the j^{th} bit of $f_a(x)$ $(1 \leq j \leq n)$ is defined to be $\sum_{i=1}^{n} x_i a_{j+i-1} \bmod 2$.

A substantial amount of research [13, 20, 24, 41, 48, 50] deals with the construction of *efficient* hash functions. This line of work contains constructions that obey weaker definitions on function families than pair-wise independence and in particular contains constructions of ϵ-AXU$_2$ functions. Unfortunately, these functions were designed to be especially efficient when their output is substantially smaller than their input (since, they were mainly brought up in the context of authentication) which is not true in our case (but is relevant in Section 7). An additional objective is to reduce the size of the family of hash functions (e.g., [19, 24]). In our setting the purpose of this is to reduce the key-length of the pseudo-random permutations.

6 Reducing the Distinguishing Probability

There are various circumstances where it is desirable to have a pseudo-random 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.

Let F be a pseudo-random 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 [36], 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 pseudo-random 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 pseudo-random) functions (see Figure 3 for an illustration).

Patarin [38] shows that if we take six rounds of the LR-Construction (instead of three or four), then the resulting permutation cannot be distinguished from a random permutation with advantage better than $\frac{5m^3}{2^\ell}$ (improving [36]). 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 pseudo-random functions with larger input-length, which might be a disadvantage for some applications.

In order 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

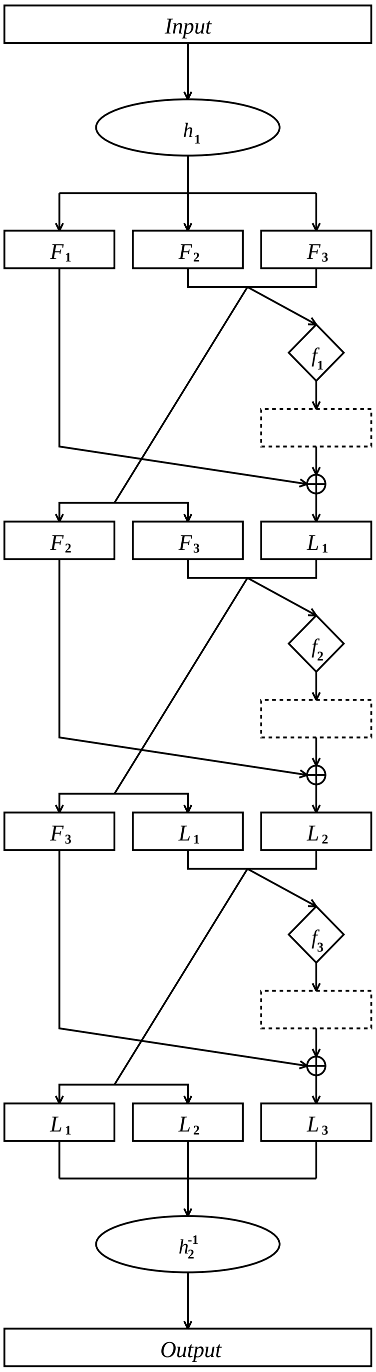

Figure 3: Construction of strong pseudo-random permutations with reduced distinguishing probability using $t+2$ rounds (here $t=3$). Recall, $f_i : I_{(1-1/t)\ell} \mapsto I_{\ell/t}$ (here $f_i : I_{2\ell/3} \mapsto I_{\ell/3}$).

length-preserving functions as in Definition 2.1). We note that using such "unbalanced" Feistel permutations was previously suggested in [5, 28, 46].

Definition 6.1 (Generalized Feistel Permutations) *For any two positive integers, s and ℓ', and any function $f : I_{\ell'} \mapsto I_s$ let $\ell = \ell' + s$ and let $\mathbf{D}_f \in P_\ell$ be the permutation defined by $\mathbf{D}_f(L, R) \stackrel{\text{def}}{=} (R, L \oplus f(R))$, where $|L| = s$ and $|R| = \ell'$.*

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.

Definition 6.2 ($t+2$-Round Construction) *For any integers $2 \le t \le \ell$, let s and r be integers such that $\ell = s \cdot t + r$ (where $r < t$). For any $h_1, h_2 \in P_\ell$, $f_1, f_2, \ldots, f_r : I_{\ell-s-1} \mapsto I_{s+1}$ and $f_{r+1}, \ldots, f_t : I_{\ell-s} \mapsto I_s$ define*

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

and

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

(We get the construction of Definition 3.1 by choosing $t = 2$, $s = \ell/2$ and $r = 0$.)

Theorem 6.1 *Let W and S be as in Definition 6.2, where h_1 & h_2 are pair-wise independent permutations and $f_1, f_2, \ldots, f_t$ are pseudo-random functions (t is allowed to be a function of ℓ); h_1, h_2 and $f_1, f_2, \ldots, f_t$ are independently chosen. Then, W is a pseudo-random permutation and S a strong pseudo-random permutation.*

Furthermore, assume that no efficient oracle-machine that makes at most $m = m(\ell)$ queries, ϵ-distinguishes between the pseudo-random functions and random functions for $\epsilon = \epsilon(n)$. Then, no efficient oracle-machine that makes at most m queries to W (resp. S and S^{-1}) ϵ'-distinguishes W (resp. S) from a random permutation for $\epsilon' = t \cdot \epsilon + \frac{t}{2} \cdot \frac{m^2}{2^{\ell - \lceil \ell/t \rceil}} + \frac{m^2}{2^\ell}$

In case the middle functions are truly random this reduces to:

Theorem 6.2 *Let S be as in Definition 6.2, 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:*

$$\left| \Pr[M^{S,S^{-1}}(1^\ell) = 1] - \Pr[M^{R,R^{-1}}(1^\ell) = 1] \right| \le \frac{t}{2} \cdot \frac{m^2}{2^{\ell - \lceil \ell/t \rceil}} + \frac{m^2}{2^\ell}$$

The proof of Theorem 6.2 follows the framework described in Section 4. Assume for simplicity that $\ell = s \cdot t$, the set $\mathrm{BAD}(h_1, h_2)$ (Definition 3.5) is replaced with the set $\mathrm{BAD}_2(h_1, h_2)$ defined 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 there exist $1 \le i < j \le m$ and $1 \le k \le t$ such that*
>
> $$(F_i^{k+1}, \ldots, F_i^t, L_i^1, \ldots, L_i^{k-1}) = (F_j^{k+1}, \ldots, F_j^t, L_j^1, \ldots, L_j^{k-1}),$$
>
> *where $(F_i^1, F_i^2, \ldots, F_i^t) = h_1(x_i)$ and $(L_i^1, L_i^2, \ldots, L_i^t) = h_2(y_i)$ ($|F_i^1| = |F_i^2| = \ldots = |F_i^t| = |L_i^1| = |L_i^2| = \ldots = |L_i^t| = s$).*

This guarantees that for any possible and consistent M-transcript σ we have that $\Pr_S[T_S = \sigma \mid \sigma \notin \mathrm{BAD}_2(h_1, h_2)] = 2^{-\ell m}$ (and hence, it is a proper definition according to the framework). The reason is that, under the notation above,

$$\forall i, y_i = S(x_i) \Longleftrightarrow \forall 1 \le i \le m,\ \forall 1 \le k \le t,\ f_k(F_i^{k+1}, \ldots, F_i^t, L_i^1, \ldots, L_i^{k-1}) = F_i^k \oplus \mathrm{L}_i^k$$

Therefore, given any specific choice of h_1 and h_2 (in the definition of S) such that $\sigma \notin \mathrm{BAD}_2(h_1, h_2)$ the event $T_S = \sigma$ is composed of $m \cdot t$ independent events each of which has probability 2^{-s} to happen. In order to apply Theorem 4.2, it remains to note that for any such σ we have that

$$\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}_2(h_1, h_2)] < t \cdot \binom{m}{2} \cdot 2^{-(\ell - \lceil \ell/t \rceil)} < \frac{t}{2} \cdot \frac{m^2}{2^{\ell - \lceil \ell/t \rceil}}$$

Remark 6.1 *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 pseudo-random 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 $\frac{\ell \cdot m^2}{2^\ell}$.*

7 SPPE on Many Blocks Using PFE or PPE on a Single Block

Consider the application of pseudo-random permutations to encryption, i.e., using $f(M)$ in order to encrypt a message M, where f is a pseudo-random 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 pseudo-random permutations or pseudo-random functions (if we want to employ the LR-Construction) with large input-length is expensive. Therefore, we would like a way to construct pseudo-random permutations (or functions) on *many blocks* from pseudo-random permutations (or functions) on *a single block*.

Several such constructions were suggested in the context of DES (see e.g. [11] 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 pseudo-random 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 pseudo-random). This may leak information about the message being encrypted (see further discussion in Section 7.2).

In this section we consider a generalization of the construction of Section 3 that uses pseudo-random functions (or permutations) on a *single* block to construct strong pseudo-random permutations on *many* 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 pseudo-random permutation) on each sub-block separately, finally, apply a second pair-wise independent permutation on the entire value you get (see Figure 4 for an illustration).

This solution resembles the ECB-mode, it is almost as simple and it is highly suitable for parallel implementation. Contrary to the ECB-mode, this construction does give a pseudo-random permutation on *the entire message* (though the security parameter is still relative to the length of a sub-block).

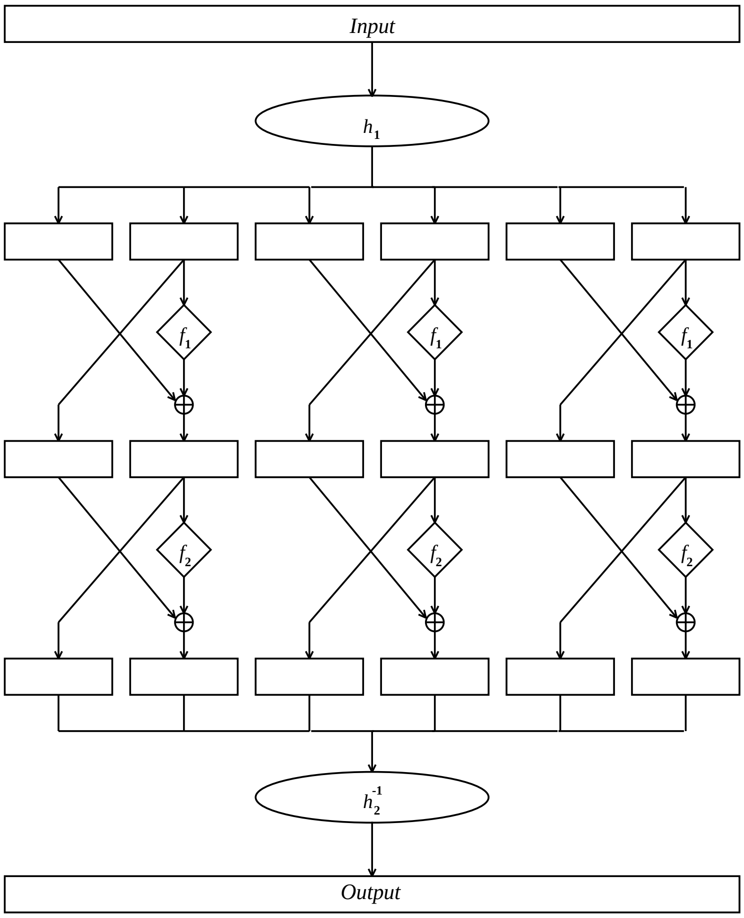

Figure 4: Construction of a strong pseudo-random permutation on many (six in this case) blocks from a pseudo-random function on a single block.

For simplicity, we only describe the construction using truly random functions (or a truly random permutation). The analysis of the construction when pseudo-random functions are used follows easily. In addition, we restrict our attention to the construction of *strong* pseudo-random permutations.

Definition 7.1 *For any two 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:

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

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 7.1 *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 7.1) 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:*

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

and

$$\left| \Pr[M^{\hat{S},\hat{S}^{-1}}(1^{2nb}) = 1] - \Pr[M^{R,R^{-1}}(1^{2nb}) = 1] \right| \leq \frac{m^2 \cdot b^2}{2^{2n-1}}$$

The proof of Theorem 7.1 for S follows the framework described in Section 4. The set $\text{BAD}(h_1, h_2)$ (Definition 3.5) is replaced with the set $\text{BAD}_3(h_1, h_2)$ defined to be:

> *The set of all possible and consistent M-transcripts, $\sigma = \{\langle x_1, y_1 \rangle, \ldots, \langle x_m, y_m \rangle\}$, such that either there are two equal values in $\{F_i^{2j}\}_{1 \leq i \leq m,\ 1 \leq j \leq b}$ or there are two equal values in $\{L_i^{2j-1}\}_{1 \leq i \leq m,\ 1 \leq j \leq b}$, where $(F_i^1, F_i^2, \ldots, F_i^{2b}) = h_1(x_i)$ and $(L_i^1, L_i^2, \ldots, L_i^{2b}) = h_2(y_i)$ ($|F_i^1| = |F_i^2| = \ldots = \left|F_i^{2b}\right| = |L_i^1| = |L_i^2| = \ldots = \left|L_i^{2b}\right| = n$).*

This guarantees that for any possible and consistent M-transcript σ we have that

$$\Pr_S[T_S = \sigma \mid \sigma \notin \text{BAD}_3(h_1, h_2)] = 2^{-2n \cdot b \cdot m}$$

(and hence, it is a proper definition according to the framework). The reason is that, under the notation above,

$$\forall i, y_i = S(x_i) \Longleftrightarrow \forall 1 \leq i \leq m,\ \forall 1 \leq j \leq b,\ f_1(F_i^{2j}) = F_i^{2j-1} \oplus L_i^{2j-1} \text{ and } f_2(L_i^{2j-1}) = F_i^{2j} \oplus L_i^{2j}$$

Therefore, given any specific choice of h_1 and h_2 (in the definition of S) such that $\sigma \notin \text{BAD}_3(h_1, h_2)$ the event $T_S = \sigma$ is composed of $2m \cdot b$ independent events each of which has probability 2^{-n} to happen. In order to apply Theorem 4.2, it remains to note that for any such σ we have that

$$\Pr_{h_1, h_2}[\sigma \in \text{BAD}_3(h_1, h_2)] \leq 2 \cdot \binom{m \cdot b}{2} \cdot 2^{-n} < \frac{m^2 \cdot b^2}{2^n}$$

The proof of Theorem 7.1 for $\hat{S}$ slightly deviates from the framework described in Section 4 (providing yet another evidence to the claim that "nobody is perfect"). The set $\text{BAD}(h_1, h_2)$ (Definition 3.5) is replaced with the set $\text{BAD}_4(h_1, h_2)$ defined to be:

The set of all possible and consistent M-transcripts, $\sigma = \{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$, such that either there are two equal values in $\{F_i^j\}_{1\le i\le m,\ 1\le j\le b}$ or there are two equal values in $\{L_i^j\}_{1\le i\le m,\ 1\le j\le b}$, where $(F_i^1, F_i^2, \ldots, F_i^b) = h_1(x_i)$ and $(L_i^1, L_i^2, \ldots, L_i^b) = h_2(y_i)$ ($|F_i^1| = |F_i^2| = \ldots = \left|F_i^b\right| = |L_i^1| = |L_i^2| = \ldots = \left|L_i^b\right| = 2n$).

Now we have that for any possible and consistent M-transcript σ,

$$\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}_4(h_1,h_2)] \le 2\cdot\binom{m\cdot b}{2}\cdot 2^{-2n} < \frac{m^2\cdot b^2}{2^{2n}}$$

but now for any such σ,

$$\Pr_{\hat{S}}[T_{\hat{S}} = \sigma \mid \sigma \notin \mathrm{BAD}_4(h_1,h_2)] = \frac{2^{2n}!}{(2^{2n} - m\cdot b)!}$$

instead of $2^{-2n\cdot b\cdot m}$ as "required" by the framework. However, the difference in probabilities is rather small which result in only a minor deviation from the proof of Theorem 3.2.

7.1 Relaxing the Construction

As in Section 5.2 we would like to reduce the requirements from h_1 and h_2 in Theorem 7.1. Our main motivation in doing so is to *decrease the key-length* of the pseudo-random permutations. We would like the key-length to be of order n – the length of the small sub-blocks and *not* of order $2nb$ – the length of the complete input (in some cases we may allow a small dependence on b).

We sketch a way to redefine the distributions on h_1 and h_2 in the definition of $\hat{S}$ (almost the same ideas apply to the definition of S). The requirement these distributions have to obey is that for any possible and consistent M-transcript σ we have that $\Pr_{h_1,h_2}[\sigma \in \mathrm{BAD}_4(h_1,h_2)]$ is "small". We use the following notation: For any $2n\cdot b$-bit string $z = (z_1, z_2, \ldots, z_b)$ (such that $\forall j,\ |z_j| = 2n$) and for all $1 \le i \le b$, denote by $z_{|_i}$ the substring z_i (the i^{th} substring of z). The requirement above can be achieved by sampling h_1 and h_2 according to a permutation distribution H such that for some small $\epsilon \ge 2^{-2n}$ we have that:

1. For any $2n\cdot b$-bit string x, $\forall 1 \le i < j \le b$, $\Pr_{h\in H}[h(x)_{|_i} = h(x)_{|_j}] \le \epsilon$ and
2. For any $2n\cdot b$-bit strings $x \ne x'$, $\forall 1 \le i,j \le b$, $\Pr_{h\in H}[h(x)_{|_i} = h(x')_{|_j}] \le \epsilon$.

We start by defining a permutation distribution H' that almost achieves this: A permutation $h' = h'_{u_1,u_2}$ sampled from H' is defined by two ϵ'-AXU$_2$ functions, $u_1 : I_{2n} \mapsto I_{2n}$ and $u_2 : I_{\lceil \log b\rceil} \mapsto I_{2n}$ (see definition of ϵ-AXU$_2$ functions in Section 5.2). For any $z = (z_1, z_2, \ldots, z_b)$ (such that $\forall j,\ |z_j| = 2n$),

$$h'_{u_1,u_2}(z) \stackrel{\mathrm{def}}{=} (z_1 \oplus u_1(z_b) \oplus u_2(1), z_2 \oplus u_1(z_b) \oplus u_2(2), \ldots, z_{b-1} \oplus u_1(z_b) \oplus u_2(b-1), z_b \oplus u_2(b))$$

It is not hard to verify that:

1′ For any $2n\cdot b$-bit string x, $\forall 1 \le i < j \le b$, $\Pr_{h'\in H'}[h'(x)_{|_i} = h'(x)_{|_j}] \le \epsilon'$ and

2′ For any $2n\cdot b$-bit strings $x \ne x'$ *such that* $x_{|_b} \ne x'_{|_b}$ and for all $1 \le i,j \le b$, $\Pr_{h'\in H'}[h'(x)_{|_i} = h'(x')_{|_j}] \le \epsilon'$.

In order to eliminate the additional requirement in $(2')$ that $x_{|_b} \neq x'_{|_b}$, we define the permutation distribution H such that a permutation h sampled from H is defined to be $h' \circ \mathbf{D}_g$ (see Definition 6.1), where h' is sampled according to H' and $g : I_{2n\cdot(b-1)} \mapsto I_{2n}$ is a ϵ'-AXU$_2$ function. Using $(1')$ and $(2')$ and the fact that for any $2n \cdot b$-bit strings $x \neq x'$:

$$\Pr_g[\mathbf{D}_g(x)_{|_b} = \mathbf{D}_g(x')_{|_b}] \leq \epsilon'$$

we get that H satisfies (1) and (2) for $\epsilon = 2\epsilon'$.

Notice that the computation of a function $h \in H$ is essentially equivalent to one computation of an ϵ-AXU$_2$ function, $g : I_{2n\cdot(b-1)} \mapsto I_{2n}$, and a few additional XOR operations per block. Using efficient constructions of ϵ-AXU$_2$ functions [13, 20, 24, 41, 48, 50] we get an efficient function h. Krawczyk [24] shows a construction of $\frac{m+\ell}{2^{\ell-1}}$-AXU$_2$ functions from m bits to ℓ bits with ℓ key-bits. Using these functions we can achieve the desired goal of reducing the key-length of h to O(n) bits.

7.2 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 pseudo-random permutation[4]. Note that when the encryption of a message M is $f(M)$, for a pseudo-random permutation f, then the only information that is leaked on M is whether or not M is equal to a previously encrypted message. This is not true for DES modes of operation. 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. The ECB-mode leaks even more information – the existence of two identical sub-blocks (in two different encrypted messages or in a single message). The reason that the ECB-mode leaks so much information is that every ciphertext-block solely depends on a single plaintext-block. Our construction implies that only very little and "non-cryptographic" diffusion (the permutations h_1 and h_2) is required in order to overcome this flaw of the ECB-mode.

Bellare and Rogaway [9] show how to convert the CBC-mode in order to construct a pseudo-random permutation with large input-length (this is the only place we are aware of that explicitly refers to the problem). The amount of work in their construction is comparable with two applications of the original CBC-mode (approximately twice the work of our construction, assuming that h_1 and h_2 are relatively efficient). The security of this construction is of similar order to the security of our construction. In contrast to our construction, [9] (as well as [6, 8]) is sequential in nature.

A different approach is to define a length-preserving pseudo-random function $\tilde{F}$ on $\tilde{\ell}$ bits using a length-preserving pseudo-random function F on ℓ bits (where $\ell < \tilde{\ell}$) and then to apply our version of the LR-Construction using $\tilde{F}$ in order to get a pseudo-random *permutation* on $2 \cdot \tilde{\ell}$ bits. The function $\tilde{F}$ can be defined to be $G \circ F \circ h$ where, h is a pair-wise independent hash function from $\tilde{\ell}$ bits to ℓ bits and G a pseudo-random (bit) generator from ℓ bits to $\tilde{\ell}$ bits. This idea may be attributed in part to Carter and Wegman [50]. Anderson and Biham [5] and Lucks [28] show how to directly apply similar ideas into the LR-Construction. A comparison between this approach and our construction relies on the specific parameters of the different primitives that are used. In particular, the parameters of the pseudo-random function F vs. the pseudo-random generator

[4] However, as shown by Bellare et. al. [8], the CBC-mode does define a construction of a pseudo-random *function* with small output length. A somewhat related solution to this problem is the so called *cascade* construction that is considered by Bellare et. al. [6].

G. For instance, for this approach to be more efficient than our construction we need that one application of G would be more efficient than $\lceil \tilde{\ell}/\ell \rceil$ applications of F.

7.2.1 Reducing the Distinguishing Probability

All the constructions of a pseudo-random permutation on many blocks from a pseudo-random function (or permutation) on a single block that are described in this subsection (including ours) have the following weakness: If the length of a single block is too small (e.g., 64-bits), then the pseudo-random permutation on many blocks is very weak even when the original pseudo-random function (or permutation) is very secure (e.g., completely random). In the following few paragraphs we discuss this problem and a way to overcome it.

Consider the permutation $S = S(h_1, f_1, f_2, h_2)$ (as in Definition 7.1), where $h_1, h_2 \in P_{2nb}$ are pair-wise independent permutations and $f_1, f_2 \in F_n$ are random functions. Our analysis of the security of S (Theorem 7.1) fails when the number of queries that the adversary makes is $\Omega(2^{n/2}/b)$ (in fact this analysis is tight). Having $2^{n/2}/b$ large enough forces a significant restriction on n. Therefore, a natural question is whether we can improve the security of the construction. A simple Information-Theoretic argument implies that all such constructions can be distinguished from random using $O(2^n/b)$ queries. This follows from the fact that with $O(2^n/b)$ queries the adversary gets much more bits than the length of the permutation's secret-key. Hence, the distribution of the answers to these queries is statistically very different from uniform (which allows an all-powerful adversary to distinguish the permutation from random).

In order to match this bound we first note that the somewhat high distinguishing probability of S is due to its vulnerability to a birthday-attack on the length of a single block. An adversary that makes $\Omega(2^{n/2}/b)$ uniformly chosen queries to S will force a collision in the inputs to f_1 (or f_2) with a constant probability. Such a collision fails our analysis (and can indeed be used to distinguish S from uniform). The solution lies in the following observation: The problem of foiling birthday-attacks when constructing a pseudo-random permutation on *many* blocks can be reduced to the problem of foiling birthday-attacks when constructing a pseudo-random function (or permutation) on *two* blocks. We demonstrate this using the Aiello and Venkatesan [1] construction of pseudo-random *functions*.

Let $\tilde{f}_1$ and $\tilde{f}_2$ be two independent copies of the pseudo-random *functions* on $2n$ bits we get when using truly random functions on n bits in the construction of Aiello and Venkatesan. By [1] distinguishing each $\tilde{f}_i$ from a truly random function (with constant probability) requires $\Omega(2^n)$ queries. Let h_1 and $h_2 \in P_{2nb}$ be pair-wise independent permutations and let the permutation $\tilde{S} = S(h_1, \tilde{f}_1, \tilde{f}_2, h_2)$ be as in Theorem 7.1 (for the parameters $n' = 2n$ and $b' = b/2$). We now get that distinguishing $\tilde{S}$ from random (with constant probability) requires $O(2^n/b)$ queries which is optimal.

8 Constructions of k-Wise δ-Dependent Permutations

In this section, we summarize the connection between the various constructions of this paper and the task of obtaining k-wise δ-dependent permutations. As mentioned in Section 5.1, Schnorr [47] suggested using the LR-Construction with truly random functions in order to get a pseudo-random generator that is secure as long as not too many bits of its output are accessible to the adversary. This idea is further treated by Maurer and Massey [30]. Maurer [29] suggested to replace the truly random functions with what he calls locally random (or almost random) functions. In the terminology of k-wise independence these ideas can be interpreted as a way of using the

LR-Construction in order to obtain k-wise δ-dependent permutations from k-wise δ'-dependent functions (as long as k is not too large). Theorem 1 in [29] implies that

> *when k-wise δ'-dependent functions are used instead of pseudo-random functions in the LR-Construction the result is a k-wise δ-dependent permutations for $\delta = O(k^2/2^n + \delta')$.*

Similar observations apply to the different constructions of our paper as discussed in this section.

Corollary 8.1 *(to 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 k-wise δ'-dependent functions. Then, $S = S(h_1, f_1, f_2, h_2)$ (as in Definition 3.1) is a k-wise δ-dependent permutation for*

$$\delta \stackrel{\text{def}}{=} \frac{k^2}{2^n} + \frac{k^2}{2^{2n}} + 2\delta'$$

Proof: Let $S_1, S_2 \in P_{2n}$ have the following distributions:

- $S_1 = S(h_1, g_1, f_2, h_2)$, where $h_1, h_2 \in P_{2n}$ are pair-wise independent, $f_2 \in F_n$ a k-wise δ'-dependent function and $g_1 \in F_n$ a truly random function.
- $S_2 = S(h_1, g_1, g_2, h_2)$, where $h_1, h_2 \in P_{2n}$ are pair-wise independent and $g_1, g_2 \in F_n$ are truly random functions.

and let $R \in P_{2n}$ be a truly random permutation. It is enough to show that for every k strings of $2n$-bits, $x_1, x_2, \ldots, x_k$, we have:

1. $\| \langle S(x_1), S(x_2), \ldots, S(x_k)\rangle - \langle S_1(x_1), S_1(x_2), \ldots, S_1(x_k)\rangle \| \le \delta'$
2. $\| \langle S_1(x_1), S_1(x_2), \ldots, S_1(x_k)\rangle - \langle S_2(x_1), S_2(x_2), \ldots, S_2(x_k)\rangle \| \le \delta'$
3. $\| \langle S_2(x_1), S_2(x_2), \ldots, S_2(x_k)\rangle - \langle R(x_1), R(x_2), \ldots, R(x_k)\rangle \| \le \frac{k^2}{2^n} + \frac{k^2}{2^{2n}}$

The reason (3) holds is that if we define an oracle machine M such that its i^{th} query is always $(+, x_i)$ and such that

$$\begin{array}{rl} & C_M(\{\langle x_1, y_1\rangle, \langle x_2, y_2\rangle, \ldots, \langle x_k, y_k\rangle\}) = 1 \\ \Longleftrightarrow & \Pr[\langle S_2(x_1), \ldots, S_2(x_k)\rangle = \langle y_1, \ldots, y_k\rangle] < \Pr[\langle R(x_1), \ldots, R(x_k)\rangle = \langle y_1, \ldots, y_k\rangle] \end{array}$$

we get by the definition of variation distance and from Theorem 3.2 that

$$\begin{array}{rl} & \| \langle S_2(x_1), S_2(x_2), \ldots, S_2(x_k)\rangle - \langle R(x_1), R(x_2), \ldots, R(x_k)\rangle \| \\ = & \left|\Pr[M^{S_2, S_2^{-1}}(1^{2n}) = 1] - \Pr[M^{R, R^{-1}}(1^{2n}) = 1]\right| \\ \le & \dfrac{k^2}{2^n} + \dfrac{k^2}{2^{2n}} \end{array}$$

(1) and (2) hold by the definition of k-wise δ'-dependent functions. For example, if

$$\| \langle S(x_1), S(x_2), \ldots, S(x_k)\rangle - \langle S_1(x_1), S_1(x_2), \ldots, S_1(x_k)\rangle \| > \delta'$$

then we can fix $h_1, h_2 \in P_{2n}$ and $f_2 \in F_n$ in the definition of both S and S_1 such that the inequality still holds. This defines k strings of n-bits, $z_1, z_2, \ldots, z_k$, (not necessarily all different) and a function V for which:

$$\langle S(x_1), S(x_2), \ldots, S(x_k)\rangle = V(\langle f_1(z_1), f_1(z_2), \ldots, f_1(z_k)\rangle) \text{ and}$$

$$\langle S_1(x_1), S_1(x_2), \ldots, S_1(x_k)\rangle = V(\langle g_1(z_1), g_1(z_2), \ldots, g_1(z_k)\rangle)$$

We get a contradiction since for any function V:

$$\begin{aligned} & \|V(\langle f_1(z_1), f_1(z_2), \ldots, f_1(z_k)\rangle) - V(\langle g_1(z_1), g_1(z_2), \ldots, g_1(z_k)\rangle)\| \\ \le\ & \|\langle f_1(z_1), f_1(z_2), \ldots, f_1(z_k)\rangle - \langle g_1(z_1), g_1(z_2), \ldots, g_1(z_k)\rangle\| \\ \le\ & \delta' \end{aligned}$$

□

In a similar way we get the following two Corollaries from the constructions of Sections 6 & 7:

Corollary 8.2 *(to Theorem 6.2) Let S be as in Definition 6.2, where h_1 and h_2 are pair-wise independent permutations and $f_1, f_2, \ldots, f_t$ are k-wise δ'-dependent functions. Then, S is a k-wise δ-dependent permutation for*

$$\delta \stackrel{\text{def}}{=} \frac{t}{2} \cdot \frac{k^2}{2^{\ell - \lceil \ell/t \rceil}} + \frac{k^2}{2^{\ell}} + t \cdot \delta'$$

Corollary 8.3 *(to Theorem 7.1) Let $h_1, h_2 \in P_{2nb}$ be pair-wise independent permutations, let $f_1, f_2 \in F_n$ be $b \cdot k$-wise δ'-dependent functions and let $p \in P_{2n}$ be a $b \cdot k$-wise δ'-dependent 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 7.1). Then, S is a k-wise δ-dependent permutation for*

$$\delta \stackrel{\text{def}}{=} \frac{k^2 \cdot b^2}{2^n} + \frac{k^2}{2^{2nb}} + 2\delta'$$

and $\hat{S}$ is a k-wise $\hat{\delta}$-dependent permutation for

$$\hat{\delta} \stackrel{\text{def}}{=} \frac{k^2 \cdot b^2}{2^{2n-1}} + \delta'$$

By taking $t = \ell$ in Corollary 8.2 we get a simple construction of a k-wise δ-dependent permutation on ℓ bits for δ as close to $\frac{(\ell+1)\cdot k^2}{2^{\ell}}$ as we wish. This construction requires ℓ applications of k-wise δ'-dependent functions from $\ell - 1$ bits to a single bit. An interesting question is to find a simple construction of k-wise δ-dependent permutations for an *arbitrarily small* δ and an arbitrary k.

An "old" proposal by the first author (see [42, page 17]) is to apply a card shuffling procedure that requires only few rounds and is oblivious in the sense that the location of a card after each round depends on a few random decisions. The specific card shuffling for which this idea is described in [42] was suggested by Aldous and Diaconis [2]. Unfortunately, to the best of our knowledge, this procedure was never proven to give (with few rounds) an almost uniform ordering of the cards. Nevertheless, we briefly describe it in order to demonstrate the concept of an oblivious card shuffling and the way that such a procedure can be used to construct a k-wise δ-dependent permutation. Finally we describe the main idea in the definition of another oblivious card shuffling for which we can prove that only few rounds are needed.

Each round (shuffle) in a card shuffling procedure is a permutation on the locations of the N cards of a deck (i.e., a permutation on the set $[N] \stackrel{\text{def}}{=} \{1, 2, \ldots, N\}$). In the case of the Aldous and Diaconis [2] card shuffling each such permutation is defined by a uniformly chosen $N/2$-bit string, $r = r_1 r_2 \ldots r_{N/2}$. Denote this permutation by Π_r then:

$$\forall 1 \le i \le N/2, \qquad \begin{cases} \Pi_r(i) = 2i - 1 \ \ \&\ \ \Pi_r(i + N/2) = 2i & \text{if } r_i = 1 \\ \Pi_r(i) = 2i \ \ \&\ \ \Pi_r(i + N/2) = 2i - 1 & \text{otherwise} \end{cases}$$

I.e., the cards at locations i and $i + N/2$ move to locations $2i - 1$ and $2i$ and their internal order is uniformly-chosen independently of all other choices. Note that $\forall x$, evaluating $\Pi_r(x)$ or $\Pi_r^{-1}(x)$ requires the knowledge of a *single* bit of r and therefore this card shuffling is indeed oblivious.

Consider s rounds of the card shuffling described above, $\Pi^s = \Pi_{r^1,\ldots,r^s} \stackrel{\text{def}}{=} \Pi_{r^s} \circ \Pi_{r^{s-1}} \circ \ldots \circ \Pi_{r^1}$, where $\{r^1, \ldots, r^s\}$ are uniformly-distributed and independent of each other. If Π^s is of statistical distance at most δ' from a uniform permutation then we can construct a k-wise δ-dependent permutation, $\tilde{\Pi}^s$, for $\delta = \delta' + \delta''$ as follows: simply take the permutation $\tilde{\Pi}^s$ to be s rounds $\Pi_{r^s} \circ \Pi_{r^{s-1}} \circ \ldots \circ \Pi_{r^1}$ where the $s \cdot N/2$ bits of $\{r^1, \ldots, r^s\}$ are *the outputs of a $(k \cdot s)$-wise δ''-dependent Binary-function, f*. Evaluating $\tilde{\Pi}^s$ (or its inverse) at a given point consists of s invocations of f. Therefore, an interesting problem is to show that Π^s is of exponentially-small statistical distance from a uniform permutation for a small value of s. In [2] it is conjectured that this can be shown for $s = O(\log^2 N)$. While this conjecture is, to the best of our knowledge, still open we can show a different card shuffling procedure for which it can be proven that $O((\log^2 N)$ rounds are sufficient. This card shuffling is defined in a recursive manner: Split the deck into two halves (locations $\{1, \ldots, N/2\}$ and locations $\{N/2, \ldots, N\}$), apply the card shuffling (recursively) on each half of the deck and merge the two (now shuffled) halves in an almost uniform way. A permutation, M, on $[N]$ is a merge if for every i and j such that $1 \leq i < j \leq N/2$ or $N/2 + 1 \leq i < j \leq N$ we have that $M(i) < M(j)$. An oblivious (in the same meaning as above) merging procedure can also be defined recursively but since the construction is rather cumbersome we omit its description. This direction may become attractive given an efficient and simple merging procedure.

A different direction to solving the problem of constructing k-wise δ-dependent permutations is to try and generalize the algebraic construction of pair-wise independent permutations. Leonard Schulman (private communication) suggested such a generalization that yields 3-wise independent permutations. He's suggestion is to use sharply 3-transitive permutation groups. A permutation group over the set $[n] = \{1, 2, \ldots, n\}$ is a subgroup of the symmetric group S_n. A permutation group G over $[n]$ is k-transitive if for every two k-tuples $\{a_1, \ldots, a_k\}$ and $\{b_1, \ldots, b_k\}$ of distinct elements of $[n]$ there exist a permutation $\pi \in G$ such that $\forall 1 \leq i \leq k,\ \pi(a_i) = b_i$. A permutation group G over $[n]$ is sharply k-transitive if for every two such tuples there exists exactly one permutation $\pi \in G$ such that $\forall 1 \leq i \leq k,\ \pi(a_i) = b_i$. A sharply k-transitive permutation group is in particular k-wise independent and indeed the algebraic construction of pair-wise independent permutations use a sharply 2-transitive permutation group (containing all the linear permutations). Schulman suggested to use the fact that there are known constructions of sharply 3-transitive permutation groups. However, this approach cannot be generalized to larger values of k: from the classification of finite simple groups it follows that for $k \geq 6$ there are no k-transitive groups over $[n]$ other than the symmetric group S_n and the alternating group A_n and there are only few such groups for $k = 4$ and $k = 5$ (see [12, 40]). One should be careful not to interpret this as implying that for $k \geq 4$ there are no efficient algebraic constructions of k-wise independent permutations. It is however justified to deduce that for $k \geq 4$ any small family of k-wise independent permutations is not a permutation group (i.e. is not closed under composition and inverse).

9 Conclusion and Further Work

The constructions described in Sections 3 & 7 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 pseudo-random 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 6. 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 Section 8.

Acknowledgments

We thank Ran Canetti, Oded Goldreich, Joe Kilian, Kobbi Nissim and Benny Pinkas for many helpful discussions and for their diligent reading of the paper. We thank the anonymous referees for their many helpful comments. 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, *Advances in Cryptology - EUROCRYPT '96*, Lecture Notes in Computer Science, Springer-Verlag, 1996, pp. 307-320.

[2] D. Aldous and P. Diaconis, Strong uniform times and finite random walks, *Advances in Applied Mathematics*, vol. 8, 1987, pp. 69-97.

[3] 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.

[4] 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.

[5] R. Anderson and E. Biham, Two practical and provably secure block ciphers: BEAR and LION, *Proc. Fast Software Encryption*, Lecture Notes in Computer Science, vol. 1039, Springer-Verlag, 1996, pp. 113-120.

[6] M. Bellare, R. Canetti and H. Krawczyk, Pseudorandom functions revisited: the cascade construction, *Proc. 37th IEEE Symp. on Foundations of Computer Science*, 1996, pp. 514-523.

[7] M. Bellare, O. Goldreich and S. Goldwasser, Incremental cryptography: the case of hashing and signing, *Advances in Cryptology - CRYPTO '94*, Lecture Notes in Computer Science, vol. 839, Springer-Verlag, 1994, pp. 216-233.

[8] M. Bellare, J. Kilian and P. Rogaway, The security of cipher block chaining, *Advances in Cryptology - CRYPTO '94*, Lecture Notes in Computer Science, vol. 839, Springer-Verlag, 1994, pp. 341-358.

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

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

[11] G. Brassard, **Modern cryptology**, Lecture Notes in Computer Science, vol. 325, Springer-Verlag, 1988.

[12] P. J. Cameron, Finite permutation groups and finite simple groups, *Bull. London Math. Soc.*, vol. 13, 1981, pp. 1-22.

[13] L. Carter and M. Wegman, Universal hash functions, *J. of Computer and System Sciences*, vol. 18, 1979, pp. 143-154.

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

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

[16] 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).

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

[18] O. Goldreich, S. Goldwasser and S. Micali, On the cryptographic applications of random functions, *Advances in Cryptology - CRYPTO '84*, Lecture Notes in Computer Science, vol. 196, Springer-Verlag, 1985, pp. 276-288.

[19] O. Goldreich and A. Wigderson, Tiny families of functions with random properties: a quality-size trade-off for hashing, *Proc. 26th ACM Symp. on Theory of Computing*, 1994, pp. 574-583.

[20] S. Halevi and H. Krawczyk, MMH: message authentication in software in the Gbit/second rates, *Proc. Fast Software Encryption*, Lecture Notes in Computer Science, Springer-Verlag, 1997.

[21] 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.

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

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

[24] H. Krawczyk, LFSR-based hashing and authentication, *Advances in Cryptology - CRYPTO '94*, Lecture Notes in Computer Science, vol. 839, Springer-Verlag, 1994, pp. 129-139.

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

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

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

[28] S. Lucks, Faster Luby-Rackoff ciphers, *Proc. Fast Software Encryption*, Lecture Notes in Computer Science, vol. 1039, Springer-Verlag, 1996, pp. 189-203.

[29] U. M. Maurer, A simplified and generalized treatment of Luby-Rackoff pseudorandom permutation generators, *Advances in Cryptology - EUROCRYPT '92*, Lecture Notes in Computer Science, Springer-Verlag, 1992, pp. 239-255.

[30] U. M. Maurer and J. L. Massey, Local randomness in pseudorandom sequences, *J. of Cryptology*, vol. 4(2), Springer-Verlag, 1991, pp. 135-149.

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

[32] M. Naor and O. Reingold, Synthesizers and their application to the parallel construction of pseudo-random functions, *Proc. 36th IEEE Symp. on Foundations of Computer Science*, 1995, pp. 170-181.

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

[34] Y. Ohnishi, *A study on data security*, Master Thesis (in Japanese), Tohuku University, Japan, 1988.

[35] J. Patarin, Pseudorandom permutations based on the DES scheme, *Proc. of EUROCODE '90*, Lecture Notes in Computer Science, Springer-Verlag, 1991, pp. 193-204.

[36] J. Patarin, New results on pseudorandom permutation generators based on the DES scheme, *Advances in Cryptology - CRYPTO '91*, Lecture Notes in Computer Science, Springer-Verlag, 1991. pp. 301-312

[37] J. Patarin, How to construct pseudorandom and super pseudorandom permutations from one single pseudorandom function, *Advances in Cryptology - EUROCRYPT '92*, Lecture Notes in Computer Science, Springer-Verlag, 1992, pp. 256-266.

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

[39] J. Pieprzyk, How to construct pseudorandom permutations from single pseudorandom functions, *Advances in Cryptology - EUROCRYPT '90*, Lecture Notes in Computer Science, vol. 473, Springer-Verlag, 1991, pp. 140-150.

[40] D. J. S. Robinson, **A course in the theory of groups – 2nd ed.**, New York : Springer-Verlag, 1996.

[41] P. Rogaway, Bucket hashing and its application to fast message authentication, *Advances in Cryptology - CRYPTO '95*, Lecture Notes in Computer Science, vol. 963, Springer-Verlag, 1995, pp. 74-85.

[42] S. Rudich, *Limits on the provable consequences of one-way functions*, PhD Thesis, U. C. Berkeley.

[43] R. A. Rueppel, On the security of Schnorr's pseudo random generator, *Advances in Cryptology - EUROCRYPT '89*, Lecture Notes in Computer Science, Springer-Verlag, 1989, pp. 423-428.

[44] B. Sadeghiyan and J. Pieprzyk, On necessary and sufficient conditions for the construction of super pseudorandom permutations, *Abstracts of ASIACRYPT '91*, Lecture Notes in Computer Science, Springer-Verlag, 1991, pp. 194-209.

[45] B. Sadeghiyan and J. Pieprzyk, A construction for super pseudorandom permutations from a single pseudorandom function, *Advances in Cryptology - EUROCRYPT '92*, Lecture Notes in Computer Science, Springer-Verlag, 1992, pp. 267-284

[46] B. Schneier and J. Kelsey, Unbalanced Feistel networks and block cipher design, *Proc. Fast Software Encryption*, Lecture Notes in Computer Science, vol. 1039, Springer-Verlag, 1996, pp. 121-144.

[47] C. P. Schnorr, On the construction of random number generators and random function generators, *Advances in Cryptology - EUROCRYPT '88*, Lecture Notes in Computer Science, vol. 330, Springer-Verlag, 1988, pp. 225-232.

[48] D. Stinson, Universal hashing and authentication codes, *Designs, Codes and Cryptography*, vol. 4 (1994), pp. 369-380.

[49] U. V. Vazirani, *Randomness, adversaries and computation*, PhD Thesis, U. C. Berkeley, 1986.

[50] M. Wegman and L. Carter, New hash functions and their use in authentication and set equality, *J. of Computer and System Sciences,* vol. 22, 1981, pp. 265-279.

[51] A. C. Yao, Theory and applications of trapdoor functions, *Proc. 23rd IEEE Symp. on Foundations of Computer Science*, 1982, pp. 80-91.

[52] Y. Zheng, T. Matsumoto and H. Imai, Impossibility and optimality results on constructing pseudorandom permutations, *Advances in Cryptology - EUROCRYPT '89*, Lecture Notes in Computer Science, vol. 434, Springer-Verlag, 1990, pp. 412-422.