The Random Oracle Model and the Ideal Cipher Model Are Equivalent

Jean-Sébastien Coron[1], Jacques Patarin[2], and Yannick Seurin[2,3]

[1] University of Luxembourg
[2] University of Versailles
[3] Orange Labs

Abstract. The Random Oracle Model and the Ideal Cipher Model are two well known idealised models of computation for proving the security of cryptosystems. At Crypto 2005, Coron *et al.* showed that security in the random oracle model implies security in the ideal cipher model; namely they showed that a random oracle can be replaced by a block cipher-based construction, and the resulting scheme remains secure in the ideal cipher model. The other direction was left as an open problem, *i.e.* constructing an ideal cipher from a random oracle. In this paper we solve this open problem and show that the Feistel construction with 6 rounds is enough to obtain an ideal cipher; we also show that 5 rounds are insufficient by providing a simple attack. This contrasts with the classical Luby-Rackoff result that 4 rounds are necessary and sufficient to obtain a (strong) pseudo-random permutation from a pseudo-random function.

1 Introduction

Modern cryptography is about defining security notions and then constructing schemes that provably achieve these notions. In cryptography, security proofs are often relative: a scheme is proven secure, assuming that some computational problem is hard to solve. For a given functionality, the goal is therefore to obtain an efficient scheme that is secure under a well known computational assumption (for example, factoring is hard). However for certain functionalities, or to get a more efficient scheme, it is sometimes necessary to work in some idealised model of computation.

The well known *Random Oracle Model* (ROM), formalised by Bellare and Rogaway [1], is one such model. In the random oracle model, one assumes that some hash function is replaced by a publicly accessible random function (the random oracle). This means that the adversary cannot compute the result of the hash function by himself: he must query the random oracle. The random oracle model has been used to prove the security of numerous cryptosystems, and it has lead to simple and efficient designs that are widely used in practice (such as PSS [2] and OAEP [3]). Obviously, a proof in the random oracle model is not fully satisfactory, because such a proof does not imply that the scheme will remain secure when the random oracle is replaced by a concrete hash function

D. Wagner (Ed.): CRYPTO 2008, LNCS 5157, pp. 1–20, 2008.

(such as SHA-1). Numerous papers have shown artificial schemes that are provably secure in the ROM, but completely insecure when the RO is instantiated with any function family (see [7]). Despite these separation results, the ROM still appears to be a useful tool for proving the security of cryptosystems. For some functionalities, the ROM construction is actually the only known construction (for example, for non-sequential aggregate signatures [6]).

The *Ideal Cipher Model* (ICM) is another idealised model of computation, similar to the ROM. Instead of having a publicly accessible random function, one has a publicly accessible random block cipher (or ideal cipher). This is a block cipher with a κ-bit key and a n-bit input/output, that is chosen uniformly at random among all block ciphers of this form; this is equivalent to having a family of 2^κ independent random permutations. All parties including the adversary can make both encryption and decryption queries to the ideal block cipher, for any given key. As for the random oracle model, many schemes have been proven secure in the ICM [5,11,14,16]. As for the ROM, it is possible to construct artificial schemes that are secure in the ICM but insecure for any concrete block cipher (see [4]). Still, a proof in the ideal cipher model seems useful because it shows that a scheme is secure against generic attacks, that do not exploit specific weaknesses of the underlying block cipher.

A natural question is whether the random oracle model and the ideal cipher model are equivalent models, or whether one model is strictly stronger than the other. Given a scheme secure with random oracles, is it possible to replace the random oracles with a block cipher-based construction, and obtain a scheme that is still secure in the ideal cipher model? Conversely, if a scheme is secure in the ideal cipher model, is it possible to replace the ideal cipher with a construction based on functions, and get a scheme that is still secure when these functions are seen as random oracles?

At Crypto 2005, Coron *et al.* [9] showed that it is indeed possible to replace a random oracle (taking arbitrary long inputs) by a block cipher-based construction. The proof is based on an extension of the classical notion of indistinguishability, called *indifferentiability*, introduced by Maurer *et al.* in [18]. Using this notion of indifferentiability, the authors of [9] gave the definition of an "indifferentiable construction" of one ideal primitive (F) (for example, a random oracle) from another ideal primitive (G) (for example an ideal block cipher). When a construction satisfies this notion, any scheme that is secure in the former ideal model (F) remains secure in the latter model (G), when instantiated using this construction. The authors of [9] proposed a slight variant of the Merkle-Damgård construction to instantiate a random oracle (see Fig. 1). Given any scheme provably secure in the random oracle model, this construction can replace the random oracle, and the resulting scheme remains secure in the ideal cipher model; other constructions have been analysed in [8].

The other direction (constructing an ideal cipher from a random oracle) was left as an open problem in [9]. In this paper we solve this open problem and show that the Luby-Rackoff construction with 6 rounds is sufficient to instantiate an ideal cipher (see Fig. 2 for an illustration). Actually, it is easy to see that it is

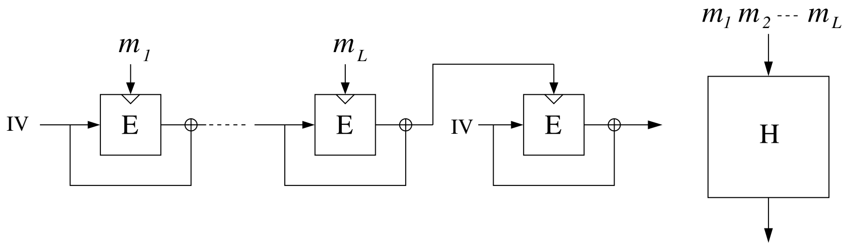


Fig. 1. A Merkle-Damgård like construction [9] based on ideal cipher E (left) to replace random oracle H (right). Messages blocks m_i's are used as successive keys for ideal-cipher E. IV is a pre-determined constant.

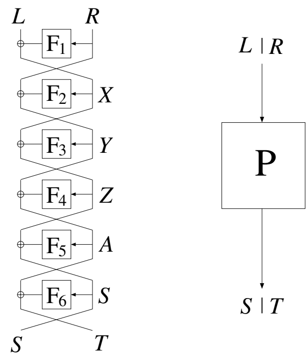


Fig. 2. The Luby-Rackoff construction with 6 rounds (left), to replace a random permutation P (right)

enough to construct a random *permutation* instead of an ideal cipher; namely, a family of 2^κ independent random permutations (*i.e.*, an ideal block cipher) can be constructed by simply prepending a k-bit key to the inner random oracle functions F_i's. Therefore in this paper, we concentrate on the construction of a random permutation. We also show that 5 rounds Luby-Rackoff is insecure by providing a simple attack; this shows that 6 rounds is actually optimal.

Our result shows that the random oracle model and the ideal cipher model are actually equivalent assumptions. It seems that up to now, many cryptographers have been reluctant to use the Ideal Cipher Model and have endeavoured to work in the Random Oracle Model, arguing that the ICM is richer and carries much more structure than the ROM. Our result shows that it is in fact not the case and that designers may use the ICM when they need it without making a stronger assumption than when working in the random oracle model. However, our security reduction is quite loose, which implies that in practice large security parameters should be used in order to replace an ideal cipher by a 6-round Luby-Rackoff.

We stress that the "indifferentiable construction" notion is very different from the classical indistinguishability notion. The well known Luby-Rackoff result that 4 rounds are enough to obtain a strong pseudo-random permutation from pseudo-random functions [17], is proven under the classical indistinguishability notion. Under this notion, the adversary has only access to the input/output of the Luby-Rackoff (LR) construction, and tries to distinguish it from a random permutation; in particular it does not have access to the input/output of the inner pseudo-random functions. On the contrary, in our setting, the distinguisher can make oracle calls to the inner round functions F_i's (see Fig. 2); the indifferentiability notion enables to accommodate these additional oracle calls in a coherent definition.

1.1 Related Work

One of the first paper to consider having access to the inner round functions of a Luby-Rackoff is [20]; the authors showed that Luby-Rackoff with 4 rounds remains secure if adversary has oracle access to the middle two round functions, but becomes insecure if adversary is allowed access to any other round functions.

In [15] a random permutation oracle was instantiated for a specific scheme using a 4-rounds Luby-Rackoff. More precisely, the authors showed that the random permutation oracle P in the Even-Mansour [14] block-cipher $E_{k_1,k_2}(m) = k_2 \oplus P(m \oplus k_1)$ can be replaced by a 4-rounds Luby-Rackoff, and the block-cipher E remains secure in the random oracle model; for this specific scheme, the authors obtained a (much) better security bound than our general bound in this paper.

In [12], Dodis and Puniya introduced a different model for indifferentiability, called indifferentiability in the *honest-but-curious* model. In this model, the distinguisher is not allowed to make direct calls to the inner hash functions; instead he can only query the global Luby-Rackoff construction and get all the intermediate results. The authors showed that in this model, a Luby-Rackoff construction with a super-logarithmic number of rounds can replace an ideal cipher. The authors also showed that indifferentiability in the honest-but-curious model implies indifferentiability in the general model, for LR constructions with up to a logarithmic number of rounds. But because of this gap between logarithmic and super-logarithmic, the authors could not conclude about general indifferentiability of Luby-Rackoff constructions. Subsequent work by Dodis and Puniya [13] studied other properties (such as unpredictability and verifiablity) of the Luby-Rackoff construction when the intermediate values are known to the attacker.

We have an observation about indifferentiability in the honest-but-curious model: general indifferentiability does not necessarily imply indifferentiability in the honest-but-curious model. More precisely, we show in Appendix B that LR constructions with up to logarithmic number of rounds are *not* indifferentiable from a random permutation in the honest-but-curious model, whereas our main result in this paper is that 6-rounds LR is indifferentiable from a random permutation in the general model.

2 Definitions

In this section, we recall the notion of indifferentiability of random systems, introduced by Maurer *et al.* in [18]. This is an extension of the classical notion of indistinguishability, where one or more oracles are publicly available, such as random oracles or ideal ciphers.

We first motivate why such an extension is actually required. The classical notion of indistinguishability enables to argue that if some system S_1 is indistinguishable from some other system S_2 (for any polynomially bounded attacker), then any application that uses S_1 can use S_2 instead, without any loss of security; namely, any non-negligible loss of security would precisely be a way of distinguishing between the two systems. Since we are interested in replacing a random permutation (or an ideal cipher) by a Luby-Rackoff construction, we would like to say that the Luby-Rackoff construction is "indistinguishable" from a random permutation. However, when the distinguisher can make oracle calls to the inner round functions, one cannot say that the two systems are "indistinguishable" because they don't even have the same interface (see Fig. 2); namely for the LR construction the distinguisher can make oracle calls to the inner functions F_i's, whereas for the random permutation he can only query the input and receive the output and vice versa. This contrasts with the setting of the classical Luby-Rackoff result, where the adversary has only access to the input/output of the LR construction, and tries to distinguish it from a random permutation. Therefore, an extension of the classical notion of indistinguishability is required, in order to show that some ideal primitive (like a random permutation) can be constructed from another ideal primitive (like a random oracle).

Following [18], we define an *ideal primitive* as an algorithmic entity which receives inputs from one of the parties and delivers its output immediately to the querying party. The ideal primitives that we consider in this paper are random oracles and random permutations (or ideal ciphers). A *random oracle* [1] is an ideal primitive which provides a random output for each new query. Identical input queries are given the same answer. A *random permutation* is an ideal primitive that contains a random permutation $P : \{0,1\}^n \to \{0,1\}^n$. The ideal primitive provides oracle access to P and P^{-1}. An *ideal cipher* is an ideal primitive that models a random block cipher $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$. Each key $k \in \{0,1\}^\kappa$ defines a random permutation $E_k = E(k,\cdot)$ on $\{0,1\}^n$. The ideal primitive provides oracle access to E and E^{-1}; that is, on query $(0,k,m)$, the primitive answers $c = E_k(m)$, and on query $(1,k,c)$, the primitive answers m such that $c = E_k(m)$. These oracles are available for any n and any κ.

The notion of indifferentiability [18] is used to show that an ideal primitive $\mathcal{P}$ (for example, a random permutation) can be replaced by a construction C that is based on some other ideal primitive $\mathcal{F}$ (for example, C is the LR construction based on a random oracle F):

Definition 1 ([18]). *A Turing machine C with oracle access to an ideal primitive $\mathcal{F}$ is said to be $(t_D, t_S, q, \varepsilon)$-indifferentiable from an ideal primitive $\mathcal{P}$ if there exists a simulator S with oracle access to $\mathcal{P}$ and running in time at most t_S, such that for any distinguisher D running in time at most t_D and making at most q queries, it holds that:*

$$\left|\Pr\left[D^{C^{\mathcal{F}},\mathcal{F}} = 1\right] - \Pr\left[D^{\mathcal{P},S^{\mathcal{P}}} = 1\right]\right| < \varepsilon$$

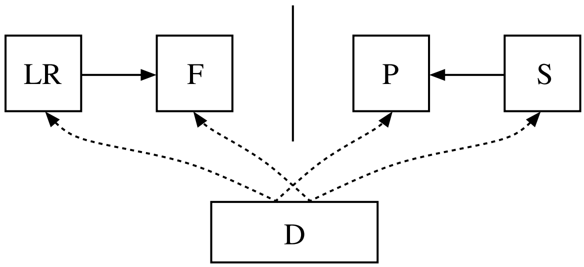


Fig. 3. The indifferentiability notion

$C^{\mathcal{F}}$ is simply said to be indifferentiable from $\mathcal{F}$ if ε is a negligible function of the security parameter n, for polynomially bounded q, t_D and t_S.

The previous definition is illustrated in Figure 3, where $\mathcal{P}$ is a random permutation, C is a Luby-Rackoff construction LR, and F is a random oracle. In this paper, for a 6-round Luby-Rackoff, we denote these random oracles $F_1, \ldots, F_6$ (see Fig. 2). Equivalently, one can consider a single random oracle F and encode in the first 3 input bits which round function $F_1, \ldots, F_6$ is actually called. The distinguisher has either access to the system formed by the construction LR and the random oracle F, or to the system formed by the random permutation P and a simulator $\mathcal{S}$. In the first system (left), the construction LR computes its output by making calls to F (this corresponds to the round functions F_i's of the Luby-Rackoff); the distinguisher can also make calls to F directly. In the second system (right), the distinguisher can either query the random permutation P, or the simulator that can make queries to P. We see that the role of the simulator is to simulate the random oracles F_i's so that no distinguisher can tell whether it is interacting with LR and F, or with P and S. In other words, 1) the output of S should be indistinguishable from that of random oracles F_i's and 2) the output of S should look "consistent" with what the distinguisher can obtain from P. We stress that the simulator does not see the distinguisher's queries to P; however, it can call P directly when needed for the simulation. Note that the two systems have the same interface, so now it makes sense to require that the two systems be indistinguishable.

To summarise, in the first system the random oracles F_i are chosen at random, and a permutation $C = LR$ is constructed from them with a 6 rounds Luby-Rackoff. In the second system the random permutation P is chosen at random and the inner round functions F_i's are simulated by a simulator with oracle access to P. Those two systems should be indistinguishable, that is the distinguisher should not be able to tell whether the inner round functions were chosen at random and then the Luby-Rackoff permutation constructed from it, or the random permutation was chosen at random and the inner round functions then "tailored" to match the permutation.

It is shown in [18] that the indifferentiability notion is the "right" notion for substituting one ideal primitive with a construction based on another ideal primitive. That is, if $C^{\mathcal{F}}$ is indifferentiable from an ideal primitive $\mathcal{P}$, then $C^{\mathcal{F}}$ can replace $\mathcal{P}$ in any cryptosystem, and the resulting cryptosystem is at least as secure in the $\mathcal{F}$ model as in the $\mathcal{P}$ model; see [18] or [9] for a proof. Our main result in this paper is that the 6 rounds Luby-Rackoff construction is indifferentiable from a random permutation; this implies that such a construction can replace a random permutation (or an ideal cipher) in any cryptosystem, and the resulting scheme remains secure in the random oracle model if the original scheme was secure in the random permutation (or ideal cipher) model.

3 Attack of Luby-Rackoff with 5 Rounds

In this section we show that 5 rounds are not enough to obtain the indifferentiability property. We do this by exhibiting for the 5 rounds Luby-Rackoff (see Fig. 4) a property that cannot be obtained with a random permutation.

Let Y and Y' be arbitrary values, corresponding to inputs of F_3 (see Fig. 4); let Z be another arbitrary value, corresponding to input of F_4. Let $Z' = F_3(Y) \oplus F_3(Y') \oplus Z$, and let:

$$X = F_3(Y) \oplus Z = F_3(Y') \oplus Z' \tag{1}$$
$$X' = F_3(Y') \oplus Z = F_3(Y) \oplus Z' \tag{2}$$

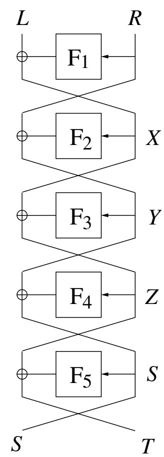


Fig. 4. 5-rounds Luby-Rackoff

From X, X', Y and Y' we now define four couples (X_i, Y_i) as follows:

$$(X_0, Y_0) = (X, Y), \quad (X_1, Y_1) = (X', Y)$$
$$(X_2, Y_2) = (X', Y'), \quad (X_3, Y_3) = (X, Y')$$

and we let $L_i \| R_i$ be the four corresponding plaintexts; we have:

$$R_0 = Y_0 \oplus F_2(X_0) = Y \oplus F_2(X)$$
$$R_1 = Y_1 \oplus F_2(X_1) = Y \oplus F_2(X')$$
$$R_2 = Y_2 \oplus F_2(X_2) = Y' \oplus F_2(X')$$
$$R_3 = Y_3 \oplus F_2(X_3) = Y' \oplus F_2(X)$$

Let Z_0, Z_1, Z_2, Z_3 be the corresponding values as input of F_4; we have from (1) and (2):

$$Z_0 = X_0 \oplus F_3(Y_0) = X \oplus F_3(Y) = Z, \quad Z_1 = X_1 \oplus F_3(Y_1) = X' \oplus F_3(Y) = Z'$$
$$Z_2 = X_2 \oplus F_3(Y_2) = X' \oplus F_3(Y') = Z, \quad Z_3 = X_3 \oplus F_3(Y_3) = X \oplus F_3(Y') = Z'$$

Finally, let $S_i \| T_i$ be the four corresponding ciphertexts; we have:

$$S_0 = Y_0 \oplus F_4(Z_0) = Y \oplus F_4(Z), \quad S_1 = Y_1 \oplus F_4(Z_1) = Y \oplus F_4(Z')$$
$$S_2 = Y_2 \oplus F_4(Z_2) = Y' \oplus F_4(Z), \quad S_3 = Y_3 \oplus F_4(Z_3) = Y' \oplus F_4(Z')$$

We obtain the relations:

$$R_0 \oplus R_1 \oplus R_2 \oplus R_3 = 0, \quad S_0 \oplus S_1 \oplus S_2 \oplus S_3 = 0$$

Thus, we have obtained four pairs (plaintext, ciphertext) such that the xor of the right part of the four plaintexts equals 0 and the xor of the left part of the four ciphertexts also equals 0. For a random permutation, it is easy to see that such a property can only be obtained with negligible probability, when the number of queries is polynomially bounded. Thus we have shown:

Theorem 1. *The Luby-Rackoff construction with* 5 *rounds is not indifferentiable from a random permutation.*

This contrasts with the classical Luby-Rackoff result, where 4 rounds are enough to obtain a strong pseudo-random permutation from pseudo-random functions.

4 Indifferentiability of Luby-Rackoff with 6 Rounds

We now prove our main result: the Luby-Rackoff construction with 6 rounds is indifferentiable from a random permutation.

Theorem 2. *The LR construction with* 6 *rounds is* $(t_D, t_S, q, \varepsilon)$*-indifferentiable from a random permutation, with* $t_S = \mathcal{O}(q^4)$ *and* $\varepsilon = 2^{18} \cdot q^8/2^n$*, where* n *is the output size of the round functions.*

Note that here the distinguisher has unbounded running time; it is only bounded to ask q queries. As illustrated in Figure 3, we must construct a simulator $\mathcal{S}$ such that the two systems formed by (LR, F) and $(P, \mathcal{S})$ are indistinguishable. The simulator is constructed in Section 4.1, while the indistinguishability property is proved in Section 4.2.

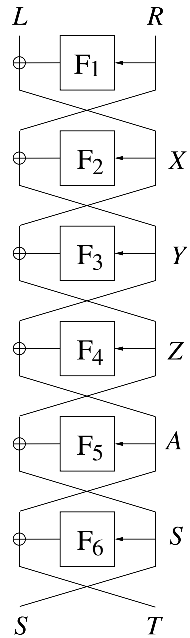
4.1 The Simulator

We construct a simulator $\mathcal{S}$ that simulates the random oracles $F_1, \ldots, F_6$. For each function F_i the simulator maintains an history of already answered queries. We write $x \in F_i$ when x belongs to the history of F_i, and we denote by $F_i(x)$ the corresponding output. When we need to obtain $F_i(x)$ and x does not belong to the history of F_i, we write $F_i(x) \leftarrow y$ to determine that the answer to F_i query x will be y; we then add $(x, F_i(x))$ to the history of F_i. We denote by n the output size of the functions F_i's. We denote by LR and LR^{-1} the 6-round Luby-Rackoff construction as obtained from the functions F_i's.

We first provide an intuition of the simulator's algorithm. The simulator must make sure that his answers to the distinguisher's F_i queries are coherent with the answers to P queries that can be obtained independently by the distinguisher. In other words, when the distinguisher makes F_i queries to the simulator (possibly in some arbitrary order), the output generated by the corresponding Luby-Rackoff must be the same as the output from P obtained independently by the distinguisher. We stress that those P queries made by the distinguisher cannot be seen by the simulator; the simulator is only allowed to make his own P queries (as illustrated in Fig. 3). In addition, the simulator's answer to F_i queries must be statistically close to the output of random functions.

The simulator's strategy is the following: when a "chain of 3 queries" has been made by the distinguisher, the simulator is going to define the values of all the other F_i's corresponding to this chain, by making a P or a P^{-1} query, so that the output of LR and the output of P are the same for the corresponding message. Roughly speaking, we say that we have a chain of 3 queries (x, y, z) when x, y, z are in the history of F_k, F_{k+1} and F_{k+2} respectively and $x = F_{k+1}(y) \oplus z$.

For example, if a query X to F_2 is received, and we have $X = F_3(Y) \oplus Z$ where Y, Z belong to the history of F_3 and F_4 respectively, then the triple (X, Y, Z) forms a 3-chain of queries. In this case, the simulator defines $F_2(X) \stackrel{\$}{\leftarrow} \{0,1\}^n$ and computes the corresponding $R = Y \oplus F_2(X)$. It also lets $F_1(R) \stackrel{\$}{\leftarrow} \{0,1\}^n$ and computes $L = X \oplus F_1(R)$. Then it makes a P-query to get $S\|T = P(L\|R)$. It also computes $A = Y \oplus F_4(Z)$. The values of $F_5(A)$ and $F_6(S)$ are then "adapted" so that the 6-round LR and the random permutation provide the same output, *i.e.* the simulator defines $F_5(A) \leftarrow Z \oplus S$ and $F_6(S) \leftarrow A \oplus T$, so that $\mathsf{LR}(L\|R) = P(L\|R) = S\|T$. In summary, given a F_2 query, the simulator looked at the history of (F_3, F_4) and adapted the answers of (F_5, F_6).



More generally, given a query to F_k, the simulator proceeds according to Table 1 below; we denote by $+$ for looking downward in the LR construction and by $-$ for looking upward. The simulator must first simulate an additional call to F_i (column "Call"). Then the simulator can compute either $L\|R$ or $S\|T$ (as determined in column "Compute"). Given $L\|R$ (resp. $S\|T$) the simulator makes a P-query (resp. a P^{-1}-query) to obtain $S\|T = P(L\|R)$ (resp. $L\|R = P^{-1}(S\|T)$). Finally Table 1 indicates the index j for which the output of (F_j, F_{j+1}) is adapted (column "Adapt").

Given a query x to F_k, with $2 \le k \le 3$, the simulator (when looking downward) must actually consider all 3-chains formed by (x, y, z) where $y \in F_{k+1}$ and $z \in F_{k+2}$. Therefore, for $k \le 2 \le 3$, one defines the following set:

$$\mathsf{Chain}(+1, x, k) = \{(y, z) \in (F_{k+1}, F_{k+2}) \mid x = F_{k+1}(y) \oplus z\}$$

where $+1$ corresponds to looking downward in the Luby-Rackoff construction. This corresponds to Lines $(F_2, +)$ and $(F_3, +)$ in Table 1.

Similarly, given a query t to F_k, with $4 \le k \le 5$, when looking upward the simulator must consider all 3-chains formed by (y, z, t) where $y \in F_{k-2}$ and $z \in F_{k-1}$; one defines the following set for $4 \le k \le 5$:

$$\mathsf{Chain}(-1, t, k) = \{(y, z) \in (F_{k-2}, F_{k-1}) \mid t = F_{k-1}(z) \oplus y\}$$

This corresponds to Lines $(F_4, -)$ and $(F_5, -)$ in Table 1.

Table 1. Simulator's behaviour

Query	Dir	History	Call	Compute	Adapt
F_1	-	(F_5, F_6)	F_4	$S\|T$	(F_2, F_3)
F_2	+	(F_3, F_4)	F_1	$L\|R$	(F_5, F_6)
F_2	-	$(\tilde{F}_6, F_1)$	F_5	$L\|R$	(F_3, F_4)
F_3	+	(F_4, F_5)	F_6	$S\|T$	(F_1, F_2)
F_4	-	(F_2, F_3)	F_1	$L\|R$	(F_5, F_6)
F_5	+	$(F_6, \tilde{F}_1)$	F_2	$S\|T$	(F_3, F_4)
F_5	-	(F_3, F_4)	F_6	$S\|T$	(F_1, F_2)
F_6	+	(F_1, F_2)	F_3	$L\|R$	(F_4, F_5)

Additionally one must consider the 3-chains obtained from a F_6 query S and looking in (F_1, F_2) history, with Line $(F_6, +)$:

$$\mathsf{Chain}(+1, S, 6) = \Big\{(R, X) \in (F_1, F_2) \mid \exists T,\ P\left(F_1\left(R\right) \oplus X \| R\right) = S \| T\Big\} \tag{3}$$

and symmetrically the 3-chains obtained from a F_1 query R and looking in (F_5, F_6) history, with Line $(F_1, -)$:

$$\mathsf{Chain}(-1, R, 1) = \Big\{(A, S) \in (F_5, F_6) \mid \exists L,\ P^{-1}(S \| F_6(S) \oplus A) = L \| R\Big\} \tag{4}$$

One must also consider the 3-chains associated with (F_1, F_6) history, obtained either from a F_2 query X or a F_5 query A, with Lines $(F_2, -)$ and $(F_5, +)$. Given a F_2 query X, we consider all $R \in F_1$, and for each corresponding $L = X \oplus F_1(R)$, we compute $S\|T = P(L\|R)$ and determine whether $S \in F_6$. Additionally, we also consider "virtual" 3-chains, where $S \notin F_6$, but S is such that $P(L'\|R') = S\|T'$ for some $(R', X') \in (F_1, F_2)$, with $L' = X' \oplus F_1(R')$ and $X' \neq X$. Formally, we denote :

$$\mathsf{Chain}(-1, X, 2) = \Big\{(R, S) \in (F_1, \tilde{F}_6) \mid \exists T,\ P(X \oplus F_1(R) \| R) = S \| T\Big\} \tag{5}$$

where $\tilde{F}_6$ in $\mathsf{Chain}(-1, X, 2)$ is defined as:

$$\tilde{F}_6 = F_6 \cup \Big\{S \mid \exists T', (R', X') \in (F_1, F_2 \setminus \{X\}), P(X' \oplus F_1(R') \| R') = S \| T'\Big\}$$

and symmetrically:

$$\mathsf{Chain}(+1, A, 5) = \Big\{(R, S) \in (\tilde{F}_1, F_6) \mid \exists L,\ P^{-1}(S \| A \oplus F_6(S)) = L \| R\Big\} \tag{6}$$

$$\tilde{F}_1 = F_1 \cup \Big\{R \mid \exists L', (A', S') \in (F_5 \setminus \{A\}, F_6),\ P^{-1}(S' \| A' \oplus F_6(S')) = L' \| R\Big\}$$

When the simulator receives a query x for F_k, it then proceeds as follows:

$\mathsf{Query}(x, k)$:

1. If x is in the history of F_k then go to step 4.
2. Let $F_k(x) \overset{\$}{\leftarrow} \{0,1\}^n$ and add $(x, F_k(x))$ to the history of F_k.
3. Call $\mathsf{ChainQuery}(x, k)$
4. Return $F_k(x)$.

The $\mathsf{ChainQuery}$ algorithm is used to handle all possible 3-chains created by the operation $F_k(x) \overset{\$}{\leftarrow} \{0,1\}^n$ at step 2:

$\mathsf{ChainQuery}(x, k)$:

1. If $k \in \{2, 3, 5, 6\}$, then for all $(y, z) \in \mathsf{Chain}(+1, x, k)$:
   (a) Call $\mathsf{CompleteChain}(+1, x, y, z, k)$.
2. If $k \in \{1, 2, 4, 5\}$, then for all $(y, z) \in \mathsf{Chain}(-1, x, k)$:
   (a) Call $\mathsf{CompleteChain}(-1, x, y, z, k)$.

The $\mathsf{CompleteChain}(b, x, y, z, k)$ works as follows: it computes the message $L\|R$ or $S\|T$ that corresponds to the 3-chain (x, y, z) given as input, without querying (F_j, F_{j+1}), where j is the index given in Table 1 (column "Adapt"). If $L\|R$ is first computed, then the simulator makes a P query to obtain $S\|T = P(L\|R)$; similarly, if $S\|T$ is first computed, then the simulator makes a P^{-1} query to obtain $L\|R = P^{-1}(S\|T)$. Eventually the output of functions (F_j, F_{j+1}) is adapted so that $\mathsf{LR}(L\|R) = S\|T$.

$\mathsf{CompleteChain}(b, x, y, z, k)$:

1. If $(b, k) = (-1, 2)$ and $z \notin F_6$, then call $\mathsf{Query}(z, 6)$, without considering in $\mathsf{ChainQuery}(z, 6)$ the 3-chain that leads to the current 3-chain (x, y, z).
2. If $(b, k) = (+1, 5)$ and $y \notin F_1$, then call $\mathsf{Query}(y, 1)$, without considering in $\mathsf{ChainQuery}(y, 1)$ the 3-chain that leads to the current 3-chain (x, y, z).
3. Given (b, k) and from Table 1:
   (a) Determine the index i of the additional call to F_i (column "Call").
   (b) Determine whether $L\|R$ or $S\|T$ must be computed first.
   (c) Determine the index j for adaptation at (F_j, F_{j+1}) (column "Adapt").
4. Call $\mathsf{Query}(x_i, i)$, where x_i is the input of F_i that corresponds to the 3-chain (x, y, z), without considering in $\mathsf{ChainQuery}(x_i, i)$ the 3-chain that leads to the current 3-chain (x, y, z).
5. Compute the message $L\|R$ or $S\|T$ corresponding to the 3-chain (x, y, z).
6. If $L\|R$ has been computed, make a P query to get $S\|T = P(L\|R)$; otherwise, make a P^{-1} query to get $L\|R = P^{-1}(S\|T)$.
7. Now all input values $(x_1, \ldots, x_6)$ to $(F_1, \ldots, F_6)$ corresponding to the 3-chain (x, y, z) are known. Additionally let $x_0 \leftarrow L$ and $x_7 \leftarrow T$.
8. If x_j is in the history of F_j or x_{j+1} is in the history of F_{j+1}, abort.
9. Define $F_j(x_j) \leftarrow x_{j-1} \oplus x_{j+1}$
10. Define $F_{j+1}(x_{j+1}) \leftarrow x_j \oplus x_{j+2}$
11. Call $\mathsf{ChainQuery}(x_j, j)$ and $\mathsf{ChainQuery}(x_{j+1}, j + 1)$, without considering in $\mathsf{ChainQuery}(x_j, j)$ and $\mathsf{ChainQuery}(x_{j+1}, j)$ the 3-chain that leads to the current 3-chain (x, y, z).

Additionally the simulator maintains an upper bound B_{max} on the size of the history of each of the F_i's; if this bound is reached, then the simulator aborts; the value of B_{max} will be determined later. This terminates the description of the simulator.

We note that all lines in Table 1 are necessary to ensure that the simulation of the F_i's is coherent with what the distinguisher can obtain independently from P. For example, if we suppress the line $(F_2, +)$ in the table, the distinguisher can make a query for Z to F_4, then Y to F_3 and $X = F_3(Y) \oplus Z$ to F_2, then $A = F_4(Z) \oplus Y$ to F_5 and since it is not possible anymore to adapt the output of (F_1, F_2), the simulator fails to provide a coherent simulation.

Our simulator makes recursive calls to the Query and $\mathsf{ChainQuery}$ algorithms. The simulator aborts when the history size of one of the F_i's is greater than B_{max}. Therefore we must prove that despite these recursive calls, this bound B_{max} is never reached, except with negligible probability, for B_{max} polynomial in the security parameter. The main argument is that the number of 3-chains in the sets $\mathsf{Chain}(b, x, k)$ that involve the P permutation (equations (3), (4), (5) and (6)), must be upper bounded by the number of P/P^{-1}-queries made by the distinguisher, which is upper bounded by q. This gives an upper bound on the number of recursive queries to F_3, F_4, which in turn implies an upper bound on the history of the other F_i's. Additionally, one must show that the simulator never aborts at Step 8 in the $\mathsf{CompleteChain}$ algorithm, except with negligible probability. This is summarised in the following lemma:

Lemma 1. *Let q be the maximum number of queries made by the distinguisher and let $B_{max} = 5q^2$. The simulator $\mathcal{S}$ runs in time $\mathcal{O}(q^4)$, and aborts with probability at most $2^{14} \cdot q^8/2^n$, while making at most $105 \cdot q^4$ queries to P or P^{-1}.*

Proof. Due to space restriction, in Appendix A we only show that the simulator's running time is $\mathcal{O}(q^4)$ and makes at most $105 \cdot q^4$ queries to P/P^{-1}. The full proof of Lemma 1 is given in the full version of this paper [10].

4.2 Indifferentiability

We now proceed to prove the indifferentiability result. As illustrated in Figure 3, we must show that given the previous simulator $\mathcal{S}$, the two systems formed by (LR, F) and $(P, \mathcal{S})$ are indistinguishable.

We consider a distinguisher $\mathcal{D}$ making at most q queries to the system (LR, F) or $(P, \mathcal{S})$ and outputting a bit γ. We define a sequence $\mathtt{Game}_0$, $\mathtt{Game}_1$, ... of modified distinguisher games. In the first game $\mathtt{Game}_0$, the distinguisher interacts with the system formed by the random permutation P and the previously defined simulator $\mathcal{S}$. In the subsequent games the system is modified so that in the last game the distinguisher interacts with (LR, F). We denote by S_i the event in game i that the distinguisher outputs $\gamma = 1$.

$\mathtt{Game}_0$: the distinguisher interacts with the simulator $\mathcal{S}$ and the random permutation P.

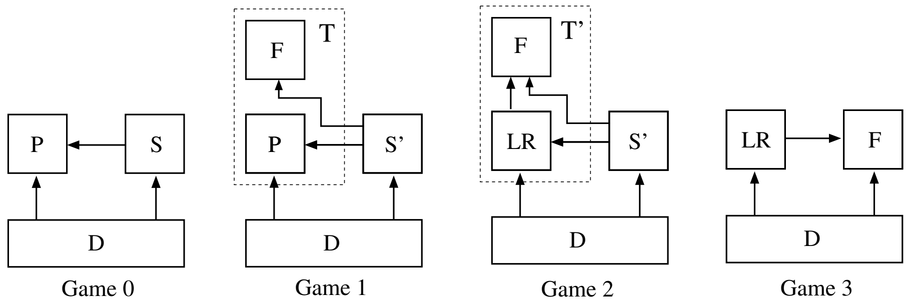


Fig. 5. Sequence of games for proving indifferentiability

$\mathtt{Game}_1$: we make a minor change in the way F_i queries are answered by the simulator, to prepare a more important step in the next game. In $\mathtt{Game}_0$ we have that a F_i query for x can be answered in two different ways: either $F_i(x) \xleftarrow{\$} \{0,1\}$, or the value $F_i(x)$ is "adapted" by the simulator. In $\mathtt{Game}_1$, instead of letting $F_i(x) \xleftarrow{\$} \{0,1\}$, the new simulator $\mathcal{S}'$ makes a query to a random oracle F_i which returns $F_i(x)$; see Fig. 5 for an illustration. Since we have simply replaced one set of random variables by a different, but identically distributed, set of random variables, we have:

$$\Pr[S_0] = \Pr[S_1]$$

$\mathtt{Game}_2$: we modify the way P and P^{-1} queries are answered. Instead of returning $P(L\|R)$ with random permutation P, the system returns $\mathsf{LR}(L\|R)$ by calling the random oracles F_i's (and similarly for P^{-1} queries).

We must show that the distinguisher's view has statistically close distribution in $\mathtt{Game}_1$ and $\mathtt{Game}_2$. For this, we consider the subsystem $\mathcal{T}$ with the random permutation P/P^{-1} and the random oracles F_i's in $\mathtt{Game}_1$, and the subsystem $\mathcal{T}'$ with Luby-Rackoff LR and random oracle F_i's in $\mathtt{Game}_2$ (see Fig. 5). We show that the output of systems $\mathcal{T}$ and $\mathcal{T}'$ is statistically close; this in turn shows that the distinguisher's view has statistically close distribution in $\mathtt{Game}_1$ and $\mathtt{Game}_2$.[1]

In the following, we assume that the distinguisher eventually makes a sequence of F_i-queries corresponding to all previous P/P^{-1} queries made by the distinguisher; this is without loss of generality, because from any distinguisher $\mathcal{D}$ we can build a distinguisher $\mathcal{D}'$ with the same output that satisfies this property.

The outputs to F_i queries provided by subsystem $\mathcal{T}$ in $\mathtt{Game}_1$ and by subsystem $\mathcal{T}'$ in $\mathtt{Game}_2$ are the same, since in both cases these queries are answered by random oracles F_i. Therefore, we must show that the output to P/P^{-1} queries provided by $\mathcal{T}$ and $\mathcal{T}'$ have statistically close distribution, when the outputs to F_i queries provided by $\mathcal{T}$ or $\mathcal{T}'$ are fixed.

[1] We do not claim that subsystems $\mathcal{T}$ and $\mathcal{T}'$ are indistinguishable for any possible sequence of queries (this is clearly false); we only show that $\mathcal{T}$ and $\mathcal{T}'$ have statistically close outputs for the particular sequence of queries made by the simulator and the distinguisher.

We can distinguish two types of P/P^{-1} queries to $\mathcal{T}$ or $\mathcal{T}'$:

– Type I: P/P^{-1} queries made by the distinguisher, or by the simulator during execution of the $\mathsf{CompleteChain}$ algorithm. From Lemma 1 there are at most $B_{max} + q \leq 6q^2$ such queries.
– Type II: P/P^{-1} queries made by the simulator when computing the sets $\mathsf{Chain}(+1, S, 6)$, $\mathsf{Chain}(-1, R, 1)$, $\mathsf{Chain}(+1, A, 5)$ and $\mathsf{Chain}(-1, X, 2)$, which are not of Type I. From Lemma 1 there are at most $Q_P = 105 \cdot q^4$ such queries.

We first consider Type I queries. Recall that the distinguisher is assumed to eventually make all the F_i queries corresponding to his P/P^{-1} queries; consequently at the end of the distinguisher's queries, the $\mathsf{CompleteChain}$ algorithm has been executed for all 3-chains corresponding to P/P^{-1} queries of Type I. We consider one such P query $L\|R$ (the argument for P^{-1} query is similar) of Type I. In $\mathtt{Game}_2$ the answer $S\|T$ can be written as follows:

$$(S, T) = (L \oplus r_1 \oplus r_3 \oplus r_5, R \oplus r_2 \oplus r_4 \oplus r_6) \tag{7}$$

where $r_1 = F_1(R)$, $r_2 = F_2(X)$, $r_3 = F_3(Y)$, $r_4 = F_4(Z)$, $r_5 = F_5(A)$ and $r_6 = F_6(S)$, and (X, Y, Z, A) are defined in the usual way.

Let j be the index used at steps 9 and 10 of the corresponding $\mathsf{CompleteChain}$ execution, and let x_j, x_{j+1} be the corresponding inputs. If the simulator does not abort during $\mathsf{CompleteChain}$, this implies that the values $r_j = F_j(x_j)$ and $r_{j+1} = F_{j+1}(x_{j+1})$ have not appeared before in the simulator's execution. This implies that $r_j = F_j(x_j)$ and $r_{j+1} = F_{j+1}(x_{j+1})$ have not appeared in a previous P/P^{-1}-query (since otherwise it would have been defined in the corresponding $\mathsf{CompleteChain}$ execution), and moreover $F_j(x_j)$ and $F_{j+1}(x_{j+1})$ have not been queried before to subsystem $\mathcal{T}'$. Since the values $r_j = F_j(x_j)$ and $r_{j+1} = F_{j+1}(x_{j+1})$ are defined by the simulator at steps 9 and 10 of $\mathsf{CompleteChain}$, these values will not be queried later to $\mathcal{T}'$. Therefore we have that $r_j = F_j(x_j)$ and $r_{j+1} = F_{j+1}(x_{j+1})$ are not included in the subsystem $\mathcal{T}'$ output; $\mathcal{T}'$ output can only include randoms in $(r_1, \ldots, r_{j-1}, r_{j+2}, \ldots, r_6)$. Therefore, we obtain from equation (7) that for fixed randoms $(r_1, \ldots, r_{j-1}, r_{j+2}, \ldots, r_6)$ the distribution of $S\|T = \mathsf{LR}(L\|R)$ in $\mathtt{Game}_2$ is uniform in $\{0,1\}^{2n}$ and independent from the output of previous P/P^{-1} queries.

In $\mathtt{Game}_1$, the output to query $L\|R$ is $S\|T = P(L\|R)$; since there are at most $q + B_{max} \leq 6 \cdot q^2$ Type I queries to P/P^{-1}, the statistical distance between $P(L\|R)$ and $\mathsf{LR}(L\|R)$ is at most $6 \cdot q^2/2^{2n}$. This holds for a single P/P^{-1} query of Type I. Since there are at most $6 \cdot q^2$ such queries, we obtain the following statistical distance δ between outputs of systems $\mathcal{T}$ and $\mathcal{T}'$ to Type I queries, conditioned on the event that the simulator does not abort:

$$\delta \leq 6 \cdot q^2 \cdot \frac{6 \cdot q^2}{2^{2n}} \leq \frac{36 \cdot q^4}{2^{2n}} \tag{8}$$

We now consider P/P^{-1} queries of Type II; from Lemma 1 there are at most $Q_P = 105 \cdot q^4$ such queries. We first consider the sets $\mathsf{Chain}(+1, S, 6)$

and $\mathsf{Chain}(-1, X, 2)$, and we consider a corresponding query $L \| R$ to P, where $L = F_1(R) \oplus X$. By definition this query is not of Type I, so no $\mathsf{CompleteChain}$ execution has occurred corresponding to this query. Given $(R, X) \in \mathsf{Chain}(+1, S, 6)$ or for $(R, S) \in \mathsf{Chain}(-1, X, 2)$, we let $Y = F_2(X) \oplus R$. If Y is not in the history of F_3, we let $F_3(Y) \xleftarrow{\$} \{0,1\}^n$; in this case, $Z = X \oplus F_3(Y)$ has the uniform distribution in $\{0,1\}^n$; this implies that Z belongs to the history of F_4 with probability at most $|F_4|/2^n \le 2q/2^n$. If Y belongs to the history of F_3, then we have that Z cannot be in the history of F_4, otherwise 3-chain (X, Y, Z) would already have appeared in $\mathsf{CompleteChain}$ algorithm, from Line $(F_2, +)$ and $(F_4, -)$ in Table 1. Therefore, we have that for all P queries $L \| R$ of Type II, no corresponding value of Z belongs to the history of F_4, except with probability at most $Q_P \cdot 2q/2^n$.

We now consider the sequence (L_i, R_i) of distinct P-queries of Type II corresponding to the previous sets $\mathsf{Chain}(+1, S, 6)$ and $\mathsf{Chain}(-1, X, 2)$. We must show that in $\mathtt{Game}_2$ the output (S_i, T_i) provided by $\mathcal{T}'$ has a distribution that is statistically close to uniform, when the outputs to F_i queries provided by $\mathcal{T}'$ are fixed. We consider the corresponding sequence of (Y_i, Z_i); as explained previously, no Z_i belongs to the simulator's history of F_4, except with probability at most $Q_P \cdot 2q/2^n$. We claim that $F_4(Z_i) \oplus Y_i \neq F_4(Z_j) \oplus Y_j$ for all $1 \le i < j \le Q_P$, except with probability at most $(Q_P)^2/2^n$. Namely, if $Z_i = Z_j$ for some $i < j$, then $F_4(Z_i) \oplus Y_i = F_4(Z_j) \oplus Y_j$ implies $Y_i = Y_j$, which gives $(L_i, R_i) = (L_j, R_j)$, a contradiction since we have assumed the (L_i, R_i) queries to be distinct. Moreover, for all $i < j$ such that $Z_i \neq Z_j$, we have that $F_4(Z_i) \oplus Y_i = F_4(Z_j) \oplus Y_j$ happens with probability at most 2^{-n}; since there are at most $(Q_P)^2$ such i, j, we have that $F_4(Z_i) \oplus Y_i = F_4(Z_j) \oplus Z_j$ for some $i < j$ happens with probability at most $(Q_P)^2/2^n$.

This implies that the elements $A_i = Y_i \oplus F_4(Z_i)$ are all distinct, except with probability at most $(Q_P)^2/2^n$. Therefore elements $S_i = Z_i \oplus F_5(A_i)$ are uniformly and independently distributed in $\{0,1\}^n$; this implies that elements S_i are all distinct, except with probability at most $(Q_P)^2/2^n$, which implies that elements $T_i = A_i \oplus F_6(S_i)$ are uniformly and independently distributed in $\{0,1\}^n$. For each (S_i, T_i), the statistical distance with $P(L_i \| R_i)$ in $\mathtt{Game}_1$ is therefore at most $Q_P/2^{2n}$. The previous arguments are conditioned on the event that no A_i or S_i belongs to the simulator's history for F_5 and F_6, which for each A_i or S_i happens with probability at most $B_{max}/2^n$. The reasoning for the sets $\mathsf{Chain}(-1, R, 1)$, $\mathsf{Chain}(+1, A, 5)$ is symmetric so we omit it. We obtain that the statistical distance δ_2 between the output of Type II P/P^{-1} queries in $\mathtt{Game}_1$ and $\mathtt{Game}_2$ is at most (conditioned on the event that the simulator does not abort):

$$\delta_2 \le 2 \cdot \left(\frac{Q_P \cdot 2q}{2^n} + 2 \cdot \frac{(Q_P)^2}{2^n} + \frac{(Q_P)^2}{2^{2n}} + \frac{Q_P \cdot B_{max}}{2^n} \right) \le \frac{2^{16} \cdot q^8}{2^n} \tag{9}$$

Let denote by Abort the event that the simulator aborts in $\mathtt{Game}_1$; we obtain from Lemma 1 and inequalities (8) and (9) :

$$|\Pr[S_2] - \Pr[S_1]| \le \Pr[\mathsf{Abort}] + \delta + \delta_2 \le \frac{2^{14} \cdot q^8}{2^n} + \frac{36 \cdot q^4}{2^{2n}} + \frac{2^{16} \cdot q^8}{2^n} \le \frac{2^{17} \cdot q^8}{2^n}$$

$\mathtt{Game}_3$: the distinguisher interacts with random system (LR, F). We have that system (LR, F) provides the same outputs as the system in $\mathtt{Game}_2$ except if the simulator fails in $\mathtt{Game}_2$. Namely, when the output values of (F_j, F_{j+1}) are adapted (steps 9 and 10 of $\mathsf{CompleteChain}$ algorithm), the values $F_j(x_j)$ and $F_{j+1}(x_{j+1})$ are the same as the one obtained directly from random oracles F_j and F_{j+1}, because in $\mathtt{Game}_2$ the P/P^{-1} queries are answered using $\mathsf{LR}/\mathsf{LR}^{-1}$. Let denote by Abort_2 the event that simulator aborts in Game_2; we have:

$$\Pr[\mathsf{Abort}_2] \leq \Pr[\mathsf{Abort}] + \delta + \delta_2 \leq \frac{2^{14} \cdot q^8}{2^n} + \frac{36 \cdot q^4}{2^{2n}} + \frac{2^{16} \cdot q^8}{2^n} \leq \frac{2^{17} \cdot q^8}{2^n}$$

which gives:

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[\mathsf{Abort}_2] \leq \frac{2^{17} \cdot q^8}{2^n}$$

From the previous inequalities, we obtain the following upper bound on the distinguisher's advantage:

$$|\Pr[S_3] - \Pr[S_0]| \leq \frac{2^{18} \cdot q^8}{2^n}$$

which terminates the proof of Theorem 2.

5 Conclusion and Further Research

We have shown that the 6 rounds Feistel construction is indifferentiable from a random permutation, a problem that was left open in [9]. This shows that the random oracle model and the ideal cipher model are equivalent models. A natural question is whether our security bound in $q^8/2^n$ is optimal or not. We are currently investigating:

- a better bound for 6 rounds (or more),
- best exponential attacks against 6 rounds (or more),
- other models of indifferentiability with possibly simpler proofs.

Acknowledgements. we would like to thank the anonymous referees of Crypto 2008 for their useful comments.

References

1. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
2. Bellare, M., Rogaway, P.: The exact security of digital signatures - How to sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
3. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)

4. Black, J.: The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 328–340. Springer, Heidelberg (2006)
5. Black, J., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block Cipher-Based Hash-Function Constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442. Springer, Heidelberg (2002)
6. Boneh, D., Gentry, C., Shacham, H., Lynn, B.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
7. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. In: Proceedings of the 30th ACM Symposium on the Theory of Computing, pp. 209–218. ACM Press, New York (1998)
8. Chang, D., Lee, S., Nandi, M., Yung, M.: Indifferentiable Security Analysis of Popular Hash Functions with Prefix-Free Padding. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 283–298. Springer, Heidelberg (2006)
9. Coron, J.S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård Revisited: How to Construct a Hash Function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)
10. Coron, J.S., Patarin, J., Seurin, Y.: The Random Oracle Model and the Ideal Cipher Model are Equivalent. Full version of this paper. Cryptology ePrint Archive, Report 2008/246, `http://eprint.iacr.org/`
11. Desai, A.: The security of all-or-nothing encryption: Protecting against exhaustive key search. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880. Springer, Heidelberg (2000)
12. Dodis, Y., Puniya, P.: On the Relation Between the Ideal Cipher and the Random Oracle Models. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 184–206. Springer, Heidelberg (2006)
13. Dodis, Y., Puniya, P.: Feistel Networks Made Public, and Applications. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 534–554. Springer, Heidelberg (2007)
14. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. In: Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 210–224. Springer, Heidelberg (1993)
15. Gentry, C., Ramzan, Z.: Eliminating Random Permutation Oracles in the Even-Mansour Cipher. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329. Springer, Heidelberg (2004)
16. Kilian, J., Rogaway, P.: How to protect DES against exhaustive key search (An analysis of DESX). Journal of Cryptology 14(1), 17–35 (2001)
17. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal of Computing 17(2), 373–386 (1988)
18. Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)
19. Patarin, J.: Pseudorandom Permutations Based on the DES Scheme. In: Charpin, P., Cohen, G. (eds.) EUROCODE 1990. LNCS, vol. 514, pp. 193–204. Springer, Heidelberg (1991)
20. Ramzan, Z., Reyzin, L.: On the Round Security of Symmetric-Key Cryptographic Primitives. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880. Springer, Heidelberg (2000)

A Proof of Lemma 1

We first give an upper bound on the total number of executions of algorithm $\mathsf{CompleteChain}(b, x, y, z, k)$ for $(b, k) \in \{(+1, 6), (-1, 2), (-1, 1), (+1, 5)\}$. We first consider the set:

$$\mathsf{Chain}(+1, S, 6) = \Big\{(R, X) \in (F_1, F_2) \mid \exists T,\ P\,(X \oplus F_1(R) \| R) = S \| T\Big\}$$

that generates executions of $\mathsf{CompleteChain}(+1, S, R, X, 6)$. We denote by bad_6 the event that $\mathsf{CompleteChain}(+1, S, R, X, 6)$ was called while $X \oplus F_1(R) \| R$ has not appeared in a P/P^{-1} query made by the distinguisher.

Similarly, considering the set $\mathsf{Chain}(-1, X, 2)$, we denote by bad_2 the event that $\mathsf{CompleteChain}(-1, X, R, S, 2)$ was called while $X \oplus F_1(R) \| R$ has not appeared in a P/P^{-1} query made by the distinguisher. Symmetrically, we denote by bad_1 and bad_5 the corresponding events for $\mathsf{CompleteChain}(-1, R, A, S, 1)$ and $\mathsf{CompleteChain}(+1, A, R, S, 5)$. We denote $\mathsf{bad} = \mathsf{bad}_1 \vee \mathsf{bad}_2 \vee \mathsf{bad}_5 \vee \mathsf{bad}_6$.

Lemma 2. *The total number of executions of* $\mathsf{CompleteChain}(b, x, y, z, k)$ *for* $(b, k) \in \{(+1, 6), (-1, 2), (-1, 1), (+1, 5)\}$ *is upper bounded by* q*, unless event* bad *occurs, which happens with probability at most:*

$$\Pr[\mathsf{bad}] \le \frac{5 \cdot (B_{max})^4}{2^n} \tag{10}$$

Proof. If event bad has not occurred, then the distinguisher has made a P/P^{-1} query corresponding to all pairs (x, y) in $\mathsf{CompleteChain}(b, x, y, z, k)$ for $(b, k) \in \{(+1, 6), (-1, 2), (-1, 1), (+1, 5)\}$; since the distinguisher makes at most q queries, the total number of executions is then upper bounded by q.

We first consider event bad_6 corresponding to $\mathsf{Chain}(+1, S, 6)$. If $L \| R$ with $L = X \oplus F_1(R)$ has never appeared in a P/P^{-1} query made by the distinguisher, then the probability that $P(L \| R) = S \| T$ for some T is at most 2^{-n}. For a single S query to F_6, the probability that bad_6 occurs is then at most $|F_1| \cdot |F_2|/2^n \le (B_{max})^2/2^n$. Since there has been at most B_{max} such queries to F_6, this gives:

$$\Pr[\mathsf{bad}_6] \le \frac{(B_{max})^3}{2^n}$$

Symmetrically, the same bound holds for event bad_1.

Similarly, for event bad_2, if the distinguisher has not made a query for $P(L \| R)$ where $L = X \oplus F_1(R)$, then the probability that $P(L \| R) = S \| T$ with $S \in \tilde{F_6}$ is at most $|\tilde{F_6}|/2^n$, where $|\tilde{F_6}| \le |F_6| + |F_1| \cdot |F_2| \le 2 \cdot (B_{max})^2$. For a single X query, this implies that event bad_2 occurs with probability at most $|F_1| \cdot |\tilde{F_6}|/2^n$; since there are at most B_{max} such queries, this gives:

$$\Pr[\mathsf{bad}_2] \le B_{max} \cdot \frac{|F_1| \cdot |\tilde{F_6}|}{2^n} \le \frac{2 \cdot (B_{max})^4}{2^n}$$

Symmetrically, the same bound holds for event bad_5. From the previous inequalities we obtain the required bound for $\Pr[\mathsf{bad}]$. $\square$

Lemma 3. *Taking $B_{max} = 5 \cdot q^2$, the history size of the simulator F_i's does not reach the bound B_{max}, unless event* bad *occurs, which happens with probability at most:*

$$\Pr[\mathsf{bad}] \leq \frac{2^{12} \cdot q^8}{2^n} \tag{11}$$

Proof. The 3-chains from Lines $(F_6, +)$, $(F_2, -)$, $(F_1, -)$ and $(F_5, +)$ in Table 1 are the only ones which can generate recursive calls to F_3 and F_4, since the other 3-chains from Lines $(F_2, +)$, $(F_3, +)$, $(F_4, -)$ and $(F_5, -)$ always include elements in F_3 and F_4 histories. Moreover from Lemma 2 the total number of corresponding executions of $\mathsf{CompleteChain}(b, x, y, z, k)$ where $(b, k) \in \{(+1, 6), (-1, 2), (-1, 1), (+1, 5)\}$ is upper bounded by q, unless event bad occurs. This implies that at most q recursive queries to F_3 and F_4 can occur, unless event bad occurs. Since the distinguisher himself makes at most q queries to F_3 and F_4, the total size of F_3 and F_4 histories in the simulator is upper bounded by $q + q = 2 \cdot q$.

The 3-chains from Lines $(F_2, +)$, $(F_3, +)$, $(F_4, -)$ and $(F_5, -)$ always include elements from both F_3 and F_4 histories. Therefore, the number of such 3-chains is upper bounded by $(2q)^2 = 4 \cdot q^2$. This implies that the simulator makes at most $4q^2$ recursive queries to F_1, F_2, F_5 and F_6. Therefore, taking:

$$B_{max} = 5 \cdot q^2 \tag{12}$$

we obtain that the simulator does not reach the bound B_{max}, except if event bad occurs; from inequality (10) and equation (12) we obtain (11). □

Lemma 4. *With $B_{max} = 5 \cdot q^2$, the simulator makes at most $105 \cdot q^4$ queries to P/P^{-1} and runs in time $\mathcal{O}(q^4)$.*

Proof. The simulator makes queries to P/P^{-1} when computing the four sets $\mathsf{Chain}(+1, S, 6)$, $\mathsf{Chain}(-1, R, 1)$, $\mathsf{Chain}(+1, A, 5)$ and $\mathsf{Chain}(-1, X, 2)$ and also when completing a 3-chain with $\mathsf{CompleteChain}$ algorithm. Since the history size of the F_i's is upper bounded by $B_{max} = 5 \cdot q^2$, we obtain that the number Q_P of P/P^{-1}-queries made by the simulator is at most:

$$Q_P \leq 4 \cdot (B_{max})^2 + B_{max} \leq 105 \cdot q^4 \tag{13}$$

From this we have that the simulator runs in time $\mathcal{O}(q^4)$. □

Lemma 2, 3 and 4 complete the first part of the proof. The remaining part consists in showing that the simulator never aborts at Step 8 in algorithm $\mathsf{CompleteChain}$, except with negligible probability, and appears in the full version of this paper [10].

B A Note on Indifferentiability in the Honest-but-Curious Model

In this section, we show that LR with up to logarithmic number of rounds is *not* indifferentiable from a random permutation in the honest-but-curious model

[12]; combined with our main result in the general model, this provides a separation between the two models. Note that this observation does not contradict any result formally proven in [12]; it only shows that honest-but-curious indifferentiability is not necessarily weaker than general indifferentiability.

Roughly speaking, in the honest-but-curious indifferentiability model, the distinguisher cannot query the F_i's directly. It can only make two types of queries: direct queries to the LR/LR^{-1} construction, and queries to the LR/LR^{-1} construction where in addition the intermediate results of the F_i's is provided. When interacting with the random permutation P and a simulator $\mathcal{S}$, the first type of query is sent directly to P, while the second type is sent to $\mathcal{S}$ who makes the corresponding query to P, and in addition provides a simulated transcript of intermediate F_i results. Note that the simulator $\mathcal{S}$ is not allowed to make additional queries to P apart from forwarding the queries from the distinguisher; see [12] for a precise definition.

The authors of [12] define the notion of a *transparent construction*. Roughly speaking, this is a construction C^F such that the value of random oracle $F(x)$ can be computed efficiently for any x, by making a polynomial number of queries to C^F and getting the F outputs used by C^F to answer each query. The authors show that Luby-Rackoff with up to logarithmic number of rounds is a transparent construction. Namely the authors construct an extracting algorithm E such that when given oracle access to LR and the intermediate values F_i used to compute LR, the value $F_i(x)$ can be computed for any x at any round i. We note that algorithm E does not make queries to LR^{-1}, only to LR.

Algorithm E implies that for a LR construction with up to logarithmic number of rounds, it is possible to find an input message $L\|R$ such that the value S in $S\|T = \mathsf{LR}(L\|R)$ has a predetermined value, by only making forward queries to LR; namely this is how algorithm E can obtain $F_\ell(S)$, where ℓ is the last round. But this task is clearly impossible with a random permutation P: it is infeasible to find $L\|R$ such that S in $S\|T = P(L\|R)$ has a pre-determined value while only making forward queries to P. This implies that a simulator in the honest-but-curious model will necessarily fail (recall that such a simulator only forwards queries from the distinguisher to P and cannot make his own queries to P/P^{-1}). Therefore, LR with up to logarithmic number of rounds is *not* indifferentiable from a random permutation in the honest-but-curious model. Since our main result is that LR with 6 rounds is indifferentiable from a random permutation in the general model, this provides a separation between the two models.