From Unpredictability to Indistinguishability: A Simple Construction of Pseudo-Random Functions from MACs

Extended Abstract

Moni Naor* and Omer Reingold**

Dept. of Applied Mathematics and Computer Science
Weizmann Institute of Science
Rehovot 76100, Israel
{naor,reingold}@wisdom.weizmann.ac.il

Abstract. This paper studies the relationship between *unpredictable functions* (which formalize the concept of a MAC) and pseudo-random functions. We show an efficient transformation of the former to the latter using a unique application of the Goldreich-Levin hard-core bit (taking the inner-product with a random vector r): While in most applications of the GL-bit the random vector r may be public, in our setting this is not the case. The transformation is only secure when r is secret and treated as part of the key. In addition, we consider weaker notions of unpredictability and their relationship to the corresponding notions of pseudo-randomness. Using these weaker notions we formulate the exact requirements of standard protocols for private-key encryption, authentication and identification. In particular, this implies a simple construction of a private-key encryption scheme from the standard challenge-response identification scheme.

1 Introduction

This paper studies several ways to weaken the definition of pseudo-random functions that come up naturally in applications such as message authentication and user identification. We focus on the concept of an unpredictable function and its relationship to a pseudo-random function. We also consider the notion of a random attack vs. an adaptive attack. We show that in several settings unpredictability can easily be turned into pseudo-randomness.

Pseudo-random functions were introduced by Goldreich, Goldwasser and Micali [12] and are a very well studied object in Foundations of Cryptography. A distribution of functions is pseudo-random if: (1) This distribution is efficient (i.e., it is easy to sample functions according to the distribution and to compute their value). (2) It is hard to tell apart a function sampled according to this

* Research supported by a grant from the Israel Science Foundation administered by the Israeli Academy of Sciences.

** Research supported by a Clore Scholars award.

distribution from a uniformly distributed function given an adaptive access to the function as a black-box.

Pseudo-random functions have numerous applications in practically any scenario where a large amount of randomness need to be shared or fixed (see e.g., [4, 7, 10, 13, 18, 19, 21]). In this paper we concentrate on the application to authentication (and also on the applications to identification and encryption): A pseudo-random function f_s can be used as a MAC (message authentication code) by letting the authentication tag of a message m be $f_s(m)$ (where the key, s, of f_s is also the private key of the MAC).

As discussed by Bellare, Canetti and Krawczyk [1] (see also [23]) the security of this scheme does not require the full strength of a pseudo-random function. Breaking this MAC (under the strong attack of existential forgery with a chosen message) amounts to adaptively querying f_s on chosen messages $m_1, m_2, \ldots m_{q-1}$ and then computing a pair $\langle m, f_s(m)\rangle$ for which m is different from $m_1, m_2, \ldots m_{q-1}$. As will be argued below, this might be hard even if f_s is not pseudo-random. Such a requirement is formalized by the concept of an unpredictable function:

A distribution of functions is unpredictable if: (1) This distribution is efficient. (2) For any efficient adversary that is given an adaptive black-box access to a function (sampled according to this distribution) it is hard to compute the value of the function at *any* point that was not queried *explicitly*.

Note that from this definition it follows that the range of an unpredictable function f_s must be large. The definition can be naturally extended to allow f_s with a range of arbitrary size N by requiring that (for any unqueried x) the advantage of computing $f_s(x)$ over the $1/N$ probability of a successful guess is negligible. However, in case N is small (i.e. polynomial) this definition implies that f_s is pseudo-random.[1] As an interesting analogy, consider Shamir's "unpredictable" number sequences [26]. There, given any prefix of the sequence it is hard to compute the next number. As shown by Yao [28], the unpredictability of the *bit* sequences introduced by Blum and Micali [6], implies their pseudo-randomness. Thus unpredictability and pseudo-randomness (indistinguishability) are equivalent for bit sequences but not for number sequences in general. This interesting phenomena is yet another reason for making a distinction between unpredictability and pseudo-randomness. Such a distinction has not always been made in the literature so far[2]

[1] A relaxation of an unpredictable function in the case of a small range N is the concept of an α-MAC. Informally, these are functions that their value (at any unqueried point) cannot be predicted with advantage over $1/N$ better than α (where α might not be non-negligible).

[2] In criticism to our approach one may suggest a different definition for unpredictable functions that makes them equivalent to pseudo-random functions. Such a definition would require *bit-by-bit* unpredictability of the function's output. I.e., that the *bit string* obtained by concatenating the output of the function on the queries of the distinguisher is unpredictable. However, we feel that the definition used in this paper is more "natural" and that the distinction between unpredictability and pseudo-randomness is useful.

Between Pseudo-Random Functions and Unpredictable Functions

Since for a random function with large enough range it is impossible to guess its value at any unqueried point, we have that a pseudo-random function with large enough range is unpredictable. Otherwise, the prediction algorithm can be used as a distinguisher. However, an unpredictable function need not "hide" anything about the input, and in particular may reveal the input. For instance, if g_s is an unpredictable function, then the function $\langle x, g_s(x)\rangle$ (x concatenated with $g_s(x)$) is an unpredictable function that completely reveals the input.

Using unpredictable functions instead of pseudo-random functions may lead to better efficiency. For example, Bellare, Canetti and Krawczyk [1] suggest that modeling cryptographic hash functions such as MD5 and SHA as being unpredictable is a realistic assumption. Nevertheless, pseudo-random functions are still valuable for many applications such as private-key encryption. In fact, pseudo-random functions are useful even in the context of authentication. Consider Wegman-Carter [27] based MACs. I.e., letting the authentication tag of a message m be $f_s(h(m))$ where h is a non-cryptographic hash-function (e.g., almost-universal$_2$).[3] Such MACs are a serious competitors to both CBC-MACs [3] and HMACs [1]. They are especially attractive for long messages since the cryptographic function is only applied to a much shorter string and since for some of the recent constructions of hash functions (e.g., [15, 25]) computing $h(m)$ is relatively cheap. However, in this case it is *not* enough for f_s to be unpredictable but it should also hide information about its input.

Since unpredictable functions imply one-way functions [17] they also imply full-fledged pseudo-random functions [12, 16]. However, these general constructions (from one-way functions to pseudo-random generators [16] and from pseudo-random generators to pseudo-random functions [12]) are computationally heavy. An obvious question at this point is whether it is possible to use unpredictable functions in order to construct a pseudo-random function *at low cost.* A natural construction is to apply the Goldreich-Levin hard-core bit [14] (**GL-bit**) in order to obtain a single-bit pseudo-random function using the inner-product with a random (but fixed) vector r. In other words, if $f : \{0,1\}^n \mapsto \{0,1\}^m$ is an unpredictable function, then consider $g : \{0,1\}^n \mapsto \{0,1\}$ where $g(x) = f(x) \odot r$ (and $\odot$ denotes the inner product mod 2). However, it turns out that the security of this construction is more delicate than may seem:

- If $r \in \{0,1\}^m$ is public, the result might *not* be pseudo-random.
- If $r \in \{0,1\}^m$ is kept secret (part of the key), the result is a single-bit pseudo-random function.

We find this result surprising since, as far as we are aware, this is the only application of the GL-bit that requires r to be secret.

One obvious disadvantage of this transformation is that we get a single-bit pseudo-random function. However, using the GL hard-core *functions* one can

[3] An alternative variant of the Wegman-Carter based MACs lets the authentication tag of a message m be $\langle r, h(m) \oplus f_s(r)\rangle$ for a random input r. In this case it is clear that the output of f_s should be pseudo-random.

extract more than a single bit at the cost of decreasing the security of the functions. Extracting t bits in such a way results in an exponential (roughly 2^{2t}) decrease in security. In case the unpredictable function is very secure, such a reduction might still be tolerable. In general, it is unrealistic to expect to extract more than a logarithmic number of pseudo-random bits from an unpredictable function (since a pseudo-random function with any super-logarithmic number of output bits is unpredictable). An alternative solution is to concatenate the inner product of a random vector r with the output of several unpredictable functions, i.e., to define the pseudo-random function $g_{s_1,s_2,\ldots,s_t,r}(x) = f_{s_1}(x) \odot r, f_{s_2}(x) \odot r, \ldots, f_{s_t}(x) \odot r$. Combining the two solutions might imply a sufficiently efficient and secure pseudo-random function with a large range. Moreover, there are several scenarios where a single-bit (or few-bit) pseudo-random function is needed. One such scenario (which also motivated this work) was considered by Canetti et. al. [8] for multicast authentication. In their scheme many functions are used for authentication, and the adversary might know a constant fraction of them. Therefore, letting each function be a one-bit pseudo-random function instead of an unpredictable function with a large range significantly reduces the size of the authentication tag while ensuring the security of the scheme.

Consequences

The main application of the transformation from unpredictability to indistinguishability is obviously for using efficient constructions of MACs in scenarios that require pseudo-random functions (especially when a single-bit pseudo-random function is needed as in [8]).

A recent work of Rivest [24] makes strong arguments against the validity of export regulations' distinction between MACs and encryption schemes. One may view our work as supporting such arguments since it shows that efficient (software or hardware) implementations of MACs can easily (and in low cost) be turned into implementations of encryption schemes. In fact, as shown by this paper, even functions that are designed for the standard challenge-response identification scheme can be used for encryption.

Random Attacks

Motivated by the requirements of standard protocols for identification and encryption, we consider two additional relaxations of unpredictable functions. The first is requiring that no efficient algorithm after adaptively querying the function can compute its value on a *random challenge* instead of any new point of its choice. The second relaxation is achieved by giving the adversary the output of the function on (polynomial number) of random inputs (instead of allowing it an adaptive attack). In addition, we consider the equivalent notions of indistinguishability. We use these concepts for:

- Identifying the exact requirements of standard schemes for authentication, identification and encryption.

- Showing that in the case of a random challenge, the transformation from unpredictability to indistinguishability is still secure even if the vector r is public. This transformation provides a simple construction of a private-key encryption scheme from the standard challenge-response identification scheme.
- Showing a more efficient variant for one of the constructions in [22] that achieves some notion of unpredictability (which is sufficient for the standard identification scheme).

Random attacks on function families are also natural in the context of Computational Learning-Theory [5]. In addition, it was shown in [20] how to construct a full-fledged pseudo-random function f from such a *weak* pseudo-random functions h (going through the concept of a pseudo-random synthesizer). Given that h has a large enough output and that f is defined on k-bit inputs, computing f involves $O(k/\log k)$ invocations of h. The construction of this paper completes the transformation from *weak* unpredictable functions to pseudo-random functions.

Since the function families that are suspected to be weak pseudo-random functions (e.g. those described in [5]; also see [20]) are extremely efficient, we consider it an important open question to improve the construction of pseudo-random functions from *weak* pseudo-random functions given in [20]. Alternatively, it would be interesting to design efficient authentication and encryption schemes that only use *weak* pseudo-random functions. We further consider these questions in Section 5.

Organization

In Section 3 we define unpredictable functions. In Section 4 we define the transformation from unpredictable functions to pseudo-random functions and show that it requires the vector r to be secret. In Section 5 we consider weaker notions of unpredictability and pseudo-randomness.

2 Preliminaries

In this section we include the definitions of function-ensembles and pseudo-random functions almost as they appear in [11, 21]:

2.1 Notation

- I^n denotes the set of all n-bit strings, $\{0,1\}^n$.
- U_n denotes the random variable uniformly distributed over I^n.
- Let x and y be two bit strings of equal length, then $x \oplus y$ denotes their bit-by-bit exclusive-or.
- Let x and y be two bit strings of equal length, then $x \odot y$ denotes their inner product mod 2.

2.2 Function-Ensembles and Pseudo-Random Function Ensembles

Let $\{A_n, B_n\}_{n\in\mathbb{N}}$ be a sequence of domains. A $A_n \mapsto B_n$ *function ensemble* is a sequence $F = \{F_n\}_{n\in\mathbb{N}}$ such that F_n is a distribution over the set of $A_n \mapsto B_n$ functions. $R = \{R_n\}_{n\in\mathbb{N}}$ is the *uniform* $A_n \mapsto B_n$ function ensemble if R_n is uniformly distributed over the set of $A_n \mapsto B_n$ functions.

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

Definition 1 negligible functions. A function $h : \mathbb{N} \mapsto \mathbb{R}^+$ is *negligible* if for every constant $c > 0$ and all sufficiently large n's

$$h(n) < \frac{1}{n^c}$$

Definition 2 pseudo-random function. Let $\{A_n, B_n\}_{n\in\mathbb{N}}$ be a sequence of domains. Let $F = \{F_n\}_{n\in\mathbb{N}}$ be an efficiently computable $A_n \mapsto B_n$ function ensemble and let $R = \{R_n\}_{n\in\mathbb{N}}$ be the uniform $A_n \mapsto B_n$ function ensemble. F is *pseudo-random* if for every efficient oracle-machine $\mathcal{M}$,

$$\left|\Pr[\mathcal{M}^{F_n}(1^n) = 1] - \Pr[\mathcal{M}^{R_n}(1^n) = 1]\right|$$

is negligible.

Remark. In these definitions, as well as in the other definitions of this paper, "efficient" is interpreted as "probabilistic polynomial-time" and "negligible" is interpreted as "smaller than $1/poly$". In fact, the proofs in this paper include more quantitative statements of security. For a discussion on security preserving reductions see [18].

3 Unpredictable Functions

In this section we define unpredictable functions. As described in the introduction, the motivation of this definition is the security of MACs. As an additional motivation, let us first consider an equivalent definition (that already appears in [12]) of pseudo-random functions through an interactive protocol. This protocol will also be used in Section 5 to define other weaker notions. For simplicity, we only consider $I^n \mapsto I^{\ell(n)}$ function-ensembles, where ℓ is some $\mathbb{N} \mapsto \mathbb{N}$ function.

Definition 3 indistinguishability against an adaptive attack. Let $F = \{F_n\}_{n\in\mathbb{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble and let $c \in \mathbb{N}$ be some constant. We define an interactive protocol that involves two parties, $\mathcal{D}$ and $\mathcal{V}$:

On the common input 1^n, the private input of $\mathcal{V}$ is a key s of a function f_s sampled from F_n and a uniformly distributed bit σ. The protocol is carried out in $q = n^c$ rounds. At the i^{th} round of the protocol $\mathcal{D}$ sends to $\mathcal{V}$ a point x_i and in return $\mathcal{V}$ sends to $\mathcal{D}$ the value $f_s(x_i)$. At the q^{th} round, $\mathcal{D}$ sends a point x_q which is different from $x_1, x_2, \ldots x_{q-1}$. In return, $\mathcal{V}$ sends $f_s(x_q)$ if $\sigma = 1$ and $y \in U_{\ell(n)}$ otherwise. Finally, $\mathcal{D}$ outputs a bit σ' which is its guess for σ.

F is *indistinguishable against an adaptive sample and an adaptive challenge* if for any polynomial time machine $\mathcal{D}$ and any constant $c \in \mathbb{N}$

$$\left|\Pr[\sigma' = \sigma] - \frac{1}{2}\right|$$

is negligible.

The equivalence of this definition to Definition 2 was shown in [12]. For a recent discussion on similar reductions and their security see the work of Bellare et. al. [2].

Proposition 4 *([12]) Let $F = \{F_n\}_{n \in \mathbb{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble. Then F is pseudo-random iff it is indistinguishable against an adaptive sample and an adaptive challenge.*

The definition of unpredictable functions is obtained from Definition 3 by replacing the requirement that $f_s(x_q)$ is indistinguishable from uniform with a requirement that $f_s(x_q)$ is hard to compute (i.e., is unpredictable):

Definition 5 unpredictable functions. Let $F = \{F_n\}_{n \in \mathbb{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble and let $c \in \mathbb{N}$ be some constant. We define an interactive protocol that involves two parties, $\mathcal{D}$ and $\mathcal{V}$:

On the common input 1^n, the private input of $\mathcal{V}$ is a key s of a function f_s sampled from F_n. The protocol is carried out in $q - 1$ rounds for $q = n^c$. At the i^{th} round of the protocol, $\mathcal{D}$ sends to $\mathcal{V}$ a point $x_i \in I^n$ and in return $\mathcal{V}$ sends to $\mathcal{D}$ the value $f_s(x_i)$. At the termination of the protocol, $\mathcal{D}$ outputs a point x_q which is different from $x_1, x_2, \ldots x_{q-1}$ and a string y which is its guess for $f_s(x_q)$.

F is *unpredictable against an adaptive sample and an adaptive challenge* if for any polynomial time machine $\mathcal{D}$ and any constant $c \in \mathbb{N}$

$$\Pr[y = f_s(x_q)]$$

is negligible.

The expression "F is an unpredictable function ensemble" is used as an abbreviation for "F is unpredictable against an adaptive sample and an adaptive challenge".

4 Turning Unpredictability into Indistinguishability

In this section we show how to apply the GL hard-core bit [14] in order to construct pseudo-random functions from unpredictable-functions. At first thought, one would imagine that such an application is straightforward as is the case with key-exchange protocols (if two parties that engage in a key-exchange protocol manage to agree on a key that cannot be computed by a passive eavesdropper then they can also easily get a secret random bit using the GL hard-core bit). However, as demonstrated below, this is not the case in our scenario.

Goldreich and Levin have shown that for every one-way function, t, given $t(x)$ (for a random input x) and given a random vector r it is infeasible to guess $r \odot x$ with non-negligible advantage over $1/2$. In fact, their result apply in a more general context: If given $t(x)$ it is hard to compute $f(x)$, then given $t(x)$ and r it is also hard to guess $f(x) \odot r$.

Since the GL-bit transforms hardness to compute into indistinguishability it is natural to apply it in our context: Given an unpredictable function $f : I^n \mapsto I^m$ a natural candidate for a pseudo-random function is $g_{s,r}(x) = f_s(x) \odot r$, where r is a random vector. Indeed, it is rather straightforward that for any unqueried input x it is hard to guess $f_s(x) \odot r$ for a random vector r chosen *after* x is fixed. However, this is *not* sufficient for proving that $g_{s,r}$ is pseudo-random: The distinguisher gets $g_{s,r}(x)$ on inputs x of its choice. Since this choice might *depend on* r it might be easy to guess $f_s(x) \odot r$ and to distinguish $g_{s,r}$ from random. As shown by the following example, this is exactly the case when the random string r is public:

The Counter-Example

Let $h_s : I^{3n} \mapsto I^n$ be an unpredictable function. Let f_s be the $I^{3n} \mapsto I^{3n}$ function such that for every input $x \in I^{3n}$ the string $y = f_s(x)$ is defined as follows:

- If at least n bits of x are zeroes, let $i_1, i_2, \ldots, i_n$ be the first locations of such bits. Then for every $1 \leq j \leq n$ the bit y_{i_j} equals the j^{th} bit of $h_s(x)$ and for any other location i the bit y_i is set to zero.
- If at least $2n$ bits of x are ones, let $i_1, i_2, \ldots, i_{2n}$ be the first locations of such bits. Then for every $1 \leq j \leq n$ the bits y_{i_j} and $y_{i_{j+n}}$ equal to the j^{th} bit of $h_s(x)$ and for any other location i the bit y_i is set to zero.

The function $f_s(x)$ is unpredictable since both mappings $\langle x, h_s(x)\rangle \mapsto \langle x, f_s(x)\rangle$ and $\langle x, f_s(x)\rangle \mapsto \langle x, h_s(x)\rangle$ are easy to compute (therefore a prediction-attack on f_s easily translates to a prediction-attack on h_s). However, for every $r \in I^{3n}$ and *every* s we have that $f_s(r) \odot r = 0$. Therefore, *when r is public*, the function $g_{s,r}$ can easily be distinguished from random. A distinguisher with access to a function P simply query for $P(r)$. If $P(r) = 0$ the distinguisher outputs "pseudo-random" and otherwise it outputs "random". In case $P = g_{s,r}$ (for any value of s) the distinguisher will output "pseudo-random" with probability 1 and in case P is truly random the distinguisher will output "pseudo-random" with probability $1/2$.

A Secret r Works

As shown by the example above, the $f_s(x) \odot r$ construction does not work in case r is public. We now show that this construction *does work when r is secret.* This fact is rather surprising since, as far as we are aware of, there is no other application of the GL-bit that requires r to be kept a secret.

Construction 4.1 *Let $F = \{F_n\}_{n \in \mathrm{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble. We define an efficient $I^n \mapsto I^1$ function-ensemble $G = \{G_n\}_{n \in \mathrm{N}}$ as follows:*

A key of a function sampled from G_n is a pair $\langle s, r\rangle$, where s is a key of a function f_s sampled from F_n and $r \in U_{\ell(n)}$. For every input $x \in I^n$ the value of $g_{s,r}$ on x is defined by

$$g_{s,r}(x) \stackrel{\text{def}}{=} f_s(x) \odot r$$

We still need to handle the fact that the distinguisher gets $g_{s,r}(x)$ on inputs x of its choice and that this choice might depend on r. However, in this case the dependence on r is only through values $g_{s,r}(y)$ that were previously queried by the distinguisher. It turns out that such a dependence is not as fatal.

Theorem 6. *Let $F = \{F_n\}_{n \in \mathrm{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble. Define $G = \{G_n\}_{n \in \mathrm{N}}$ as in Construction 4.1. If F is an unpredictable function ensemble then G is a pseudo-random function ensemble.*

Proof. (Sketch) Assume that there is an efficient oracle-machine $\mathcal{M}$ that distinguishes G from random with non-negligible advantage $\epsilon = \epsilon(n)$ (as in Definition 2). Let $q = q(n)$ be a polynomial bound on the number of queries made by $\mathcal{M}$. Assume wlog that $\mathcal{M}$ always makes exactly q different queries.

In order to prove the theorem it is sufficient to construct an efficient oracle machine $\mathcal{A}$ that operates as follows: on input $r \in U_{\ell(n)}$ and access to a function f_s sampled from F_n $\mathcal{A}$ first chooses an input $x \in I^n$ which only depends on its internal coin-tosses. I.e., x is *independent of r*. After making at most q queries to f_s *which are all different from x* it outputs a guess for $f_s(x) \odot r$ which is correct with probability at least $1/2 + \epsilon/q$.

To see that such a machine $\mathcal{A}$ is indeed sufficient, note that for at least $\epsilon/2q$ fraction of the choices for the internal coin-tosses of $\mathcal{A}$ the probability that it succeeds in guessing $f_s(x) \odot r$ is at least $1/2 + \epsilon/2q$. Therefore, we can now apply the Goldreich-Levin-Rackoff reconstruction algorithm[4] to get an efficient oracle machine $\mathcal{D}$ such that on input 1^n and access to a function f_s sampled from F_n operates as follows: $\mathcal{D}$ first chooses an input $x \in I^n$. After making $O(\ell(n) \cdot (q/\epsilon)^2 \cdot q)$ queries to f_s *which are all different from x* it outputs a guess for $f_s(x)$ which is correct with probability $\Omega((\epsilon/q)^2)$. This contradicts

[4] The Goldreich-Levin Theorem is a constructive one that enables reconstruction of x given an algorithm for guessing $x \odot r$. See [11] for details; the algorithm there is due to Rackoff.

the assumption that F is an unpredictable function-ensemble and completes the proof of the theorem.

It remains to define $\mathcal{A}$ that has the required properties:

The definition of $\mathcal{A}$: We assume that $\mathcal{A}$ knows whether or not $\Pr[\mathcal{M}^{F_n}(1^n) = 1] > \Pr[\mathcal{M}^{R_n}(1^n) = 1]$. This information can be given to $\mathcal{A}$ as part of the input (by $\mathcal{D}$ that can afford to try both possibilities). Another standard way that $\mathcal{A}$ can learn this information is by sampling. Assume wlog that indeed

$$\Pr[\mathcal{M}^{F_n}(1^n) = 1] > \Pr[\mathcal{M}^{R_n}(1^n) = 1] + \epsilon(n)$$

The algorithm $\mathcal{A}$ executes the following algorithm:

1. Sample $1 \leq J < q$ uniformly at random.
2. Invoke $\mathcal{M}$ on input 1^n.
3. Answer each one of the first J queries of $\mathcal{M}$ with a uniformly chosen bit. Denote by x the J^{th} query and by σ the answer given to it.
4. Let x^i be the i^{th} query for $i > J$, answer this query with $f_s(x_i) \odot r$ (by querying f_s on x_i).
5. If $\mathcal{M}$ outputs 1 then output σ. Otherwise output $\bar{\sigma}$.

It is immediate that the choice of x is indeed independent of r. Proving the success probability of $\mathcal{A}$ (claimed above) is done by a standard hybrid argument.

For any unpredictable function f_s, Construction 4.1 gives a *single-bit* pseudo-random function $g_{s,r}$. Extracting more bits is possible in two (complementary) ways:

1. Taking the inner product of the unpredictable function f_s with *a few* random vectors. I.e., using the function $\bar{g}_{s,r_1,r_2,\ldots,r_t}(x) = f_s(x) \odot r_1, f_s(x) \odot r_2, \ldots, f_s(x) \odot r_t$.
2. Taking the inner product of *any polynomial number* of (independent) unpredictable functions f_{s_t} with the same random vector. I.e., using the function $\hat{g}_{s_1,s_2,\ldots,s_t,r}(x) = f_{s_1}(x) \odot r, f_{s_2}(x) \odot r, \ldots, f_{s_t}(x) \odot r$.

While the first method is more efficient (the function f_s is only computed once) it decreases security more rapidly. More precisely, assume that there is an efficient oracle-machine $\mathcal{M}$ that distinguishes $\bar{g}_{s,r_1,r_2,\ldots,r_t}$ from random with advantage ϵ using q queries then it is possible to define an oracle machine $\mathcal{A}$ *as in the proof of Theorem 6* that outputs a guess for $f_s(x) \odot r$ which is correct with probability at least $1/2 + \epsilon/(q \cdot 2^t)$. Therefore it is possible to define a machine $\mathcal{D}$ that breaks the unpredictable function f with $O(\ell(n) \cdot (q/\epsilon)^2 \cdot 2^{2t} \cdot q)$ queries and success probability $\Omega((\epsilon/q)^2 \cdot 2^{-2t})$. However, in case f_s is sufficiently secure and t is not too large (say, $t = 20$) this method can still be used. For the second method, it is not hard to show a much more moderate reduction in security. I.e., a reduction by $1/t^2$ factor (getting a factor of $1/t$ is possible by using t different strings r_i instead of a single string r). The two methods can naturally be combined to give a reasonably efficient and secure pseudo-random function with a large output.

5 Weaker Notions

In this section we consider weaker notions of indistinguishability and unpredictability then those of Definitions 3 and 5. We show how to relax either one of these definitions by allowing the adversary a random attack rather than an adaptive attack. As will be described below, such random attacks come up naturally in applications such as identification and encryption. Two meanings in which an attack can be random are:

1. **A Random Challenge.** The adversary is required to compute the value of f_s on a random point. This is formalized by letting $\mathcal{V}$ send $x_q \in U_n$ to $\mathcal{D}$ after the first $q-1$ rounds.
2. **A Random Sample.** The adversary gets the value of f_s on polynomial number of random inputs instead of adaptively choosing the inputs itself. This is formalized by removing the first $q-1$ rounds of the protocol and adding to the common input the values $\langle x_1, f_s(x_1), x_2, f_s(x_2), \ldots x_{q-1}, f_s(x_{q-1})\rangle$, where each one of the x_i's is an independent instance of U_n.

Remark. An alternative to an adaptive attack and a random attack is a *static attack*. In this case, $\mathcal{D}$ has to choose and send $x_1, x_2, \ldots x_q$ at the first round. Such an attack seems less natural in the applications we consider here and we therefore ignore it. For some intuition on the difference between adaptive and static attacks see [21].

The total number the definitions we obtain by considering all combinations (i.e., indistinguishability vs. unpredictability, adaptive samples vs. random samples and adaptive challenges vs. random challenges) is eight. The observation that no two of these definitions are equivalent (as long as one-way functions exist) easily follows from the separations we sketch below. Furthermore, there are no implications except for the obvious ones:

- Let f_s be a pseudo-random function and define the function $g_s(x) = \langle x, f_s(x)\rangle$ (x concatenated with $f_s(x)$). Then g_s is an *unpredictable* function but is not *indistinguishable* even against a random sample and a random challenge.
- Let f_s be a pseudo-random function and define the function g_s such that $g_s(x) = f_s(x)$ for every $x \neq 0$ and $g_s(0) = 0$. Then g_s is indistinguishable against an adaptive sample and a *random challenge* but is not even unpredictable against a random sample and an *adaptive challenge*.
- Let f_s be a pseudo-random function and define the function g_s such that $g_s(x) = f_s(x)$ for every $x \neq f_s(0)$ and (unless the rare condition $f_s(0) = 0$ holds) $g_s(f_s(0)) = s$. Then g_s is indistinguishable against a *random sample* and an adaptive challenge but is not even unpredictable against an *adaptive samples* and a random challenge.

More "natural" examples for functions that are suspected to be secure (indistinguishable) against a random attack but are completely insecure against an adaptive attack come up in the context of Computational Learning-Theory (see [5, 20] for details). Consider for example the following distribution on functions with parameters k and n. Each function is defined by two, uniformly distributed, disjoint sets $A, B \subset \{1, \ldots, n\}$ each of size k. Given an n-bit input, the output of the function is the exclusive-or of two values: the parity of the bits indexed by A and the majority of the bits indexed by B. Restating [5] in the terminology of this paper, it is estimated there that distinguishing these functions (for $k = \log n$) from a random function using a *random sample and a random challenge* requires "profoundly" new ideas. However, the key of such a function (for any k) can easily be recovered using an adaptive attack.

The extreme efficiency of function families that are suspected to be *weak* pseudo-random functions (i.e., indistinguishable against a random sample and a random challenge) raises the following questions:

1. Can the construction in [20] of a full-fledged pseudo-random function from *weak* pseudo-random functions be improved?
2. Can *weak* pseudo-random functions be directly used in private-key encryption and authentication schemes?

We further consider the second question in Section 5.1.

5.1 The Requirements of Private-Key Tasks

Identifying the exact requirements for function families used in any given protocol can imply more efficient implementations of this protocol. We therefore consider in this section the actual requirements for standard private-key schemes. The three most common tasks in private-key cryptography are user identification, message authentication and encryption. Consider the following schemes for the above tasks. A group of parties that share a pseudo-random function f_s may perform:

Authentication The authentication tag of a message m is defined to be $f_s(m)$. Here the requirement is *unpredictability* against an adaptive sample and an adaptive challenge (in case we want existential unforgeability against a chosen message attack).

Identification A member of the group, $\mathcal{V}$, determines if $\mathcal{A}$ is also a member by issuing a random challenge r and verifying that the respond of $\mathcal{A}$ is $f_s(r)$. Assuming that the adversary can perform an active attack (i.e., can participate in executions of the protocol as the verifier), we need unpredictability against an adaptive sample and a *random* challenge. If the adversary is limited to a passive attack (i.e., can only eavesdrop to previous executions of the protocol), then we only need unpredictability against a *random* sample and a random challenge.

Encryption The encryption of a message m is defined to be $\langle r, f_s(r) \oplus m\rangle$, where r is a uniformly chosen input.
We are using the terminology of [9] for attacks (chosen plaintext, chosen ciphertext in the preprocessing and postprocessing modes) and notions security (semantic and non-malleability). Assuming that the adversary is limited to a chosen plaintext attack, we need indistinguishability against a *random*

sample and a *random* challenge (in case we are interested in semantic security). If the adversary can perform a *chosen ciphertext attack* in the preprocessing mode, then we need indistinguishability against an *adaptive* sample and a random challenge to get semantic security. For any implementation of f this scheme is malleable and hence not secure against a chosen ciphertext attack in the postprocessing mode. I.e., when the adversary queries the function *after getting the challenge.*

The functions used in all the schemes considered above should be secure against an *adaptive sample* (when we consider the stronger attack in each case). The following encryption scheme (that can also be used for authentication and identification) proposed in the full version of [9] eliminates this requirement. The encryption of a message m under this scheme is defined to be

$$\langle r, f(r) \oplus m, g(r, f(r) \oplus m) \rangle,$$

where r is a uniformly chosen input. To get non-malleable security against a chosen ciphertext attack in the postprocessing mode it is enough for f and g to be indistinguishable against a *random sample* and an *adaptive challenge*. The role of g is to "authenticate" the first part of the encryption and make it infeasible for an adversary to generate valid ciphertexts it did not explicitly receive (i.e. the encryption scheme is self-validating). An interesting open question is whether there exist an efficient authentication or encryption scheme which can be based on functions secure against a random sample and a random challenge.

5.2 Improving Efficiency for Weaker Definitions

In this section we give another demonstration that weaker definitions may imply better efficiency. We do so by showing a more efficient variant for one of the constructions of [22] that is sufficient for the standard identification scheme.

In [22], two related constructions of pseudo-random functions are presented. The construction that is based on factoring gives a single-bit (or few-bits) pseudo-random function. We show that if we are only interested in *unpredictability* against an adaptive sample and a *random challenge* this construction can be improved.

Informally, the construction of pseudo-random functions that are at least as secure as factoring is as follows: Let N be distributed over Blum-integers ($N = P \cdot Q$, where P and Q are primes and $P = Q = 3 \bmod 4$) and assume that (under this distribution) it is hard to factor N. Let g be a uniformly distributed quadratic residue in $\mathbb{Z}_N^*$, let $\mathbf{a} = \langle a_{1,0}, a_{1,1}, a_{2,0}, a_{2,1}, \ldots a_{n,0}, a_{n,1} \rangle$ be a uniformly distributed sequence of $2n$ elements in $[N] \stackrel{\text{def}}{=} \{1, 2, \ldots, N\}$ and let r be a uniformly distributed bit-string of the same length as N. Then the Binary-function, $f_{N,g,\mathbf{a},r}$, is pseudo-random. Where the value of $f_{N,g,\mathbf{a},r}$ on any n-bit input, $x = x_1 x_2 \cdots x_n$, is defined by:

$$f_{N,g,\mathbf{a},r}(x) \stackrel{\text{def}}{=} \left(g^{\prod_{i=1}^{n} a_{i,x_i}} \bmod N \right) \odot r$$

Using similar techniques to the proof in [22], it can be shown that if factoring Blum-integers is hard then the function $\tilde{f}_{N,g,\mathbf{a}}$, is unpredictable against an adaptive sample and a random challenge. Where the value of $\tilde{f}_{N,g,\mathbf{a}}$ on any n-bit input, $x = x_1 x_2 \cdots x_n$, is defined by:

$$\tilde{f}_{N,g,\mathbf{a}}(x) \stackrel{\text{def}}{=} g^{\prod_{i=1}^{n} a_{i,x_i}} \bmod N$$

As described in Section 5.1, such a function can be used for the standard challenge-response identification scheme.

5.3 Additional Transformations of Unpredictability to Indistinguishability

In Section 4, we considered the $g_{s,r}(x) = f_s(x) \odot r$ construction (Construction 4.1) as a transformation of unpredictable functions to pseudo-random functions. As discussed there, the problem in using a public r in this construction is that it enables the distinguisher to choose inputs for $g_{s,r}(x)$ that *directly depend on r*. For such an input x, the value $g_{s,r}(x)$ might be distinguishable from random. However, when we consider weaker definitions of unpredictability and indistinguishability where the *challenge is random* such a problem does not occur. In this case a rather simple application of the GL-bit gives the following theorem:

Theorem 7. *Let $F = \{F_n\}_{n \in \mathbb{N}}$ be an efficient $I^n \mapsto I^{\ell(n)}$ function-ensemble. Define $G = \{G_n\}_{n \in \mathbb{N}}$ as in Construction 4.1. It follows that:*

1. *If F is unpredictable against an adaptive sample and a random challenge, then G is indistinguishable against an adaptive sample and a random challenge.*
2. *If F is unpredictable against a random sample and a random challenge, then G is indistinguishable against a random sample and a random challenge.*

Both (1) and (2) hold even if for each function $g_{s,r} \in G_n$ we let r be public

As discussed in Section 5.1, *indistinguishability* against an adaptive sample and a random challenge is sufficient for the standard private-key encryption scheme whereas *unpredictability* against an adaptive sample and a random challenge is sufficient for the standard challenge-response identification scheme. Therefore, any function that is designed for the identification scheme can be transformed into a private-key encryption scheme (using the methods described in Section 4 for getting a larger output length).

6 Conclusion and Further Research

We have considered several notions of unpredictability and their relationship with the corresponding notions of indistinguishability. For three of these notions we have shown that the Goldreich-Levin hard-core bit can simply turn unpredictability into indistinguishability. By this construction efficient implementations of MACs can be used to obtain efficient implementations of pseudo-random functions. An interesting open problem is to prove or disprove the validity of the construction in a fourth setting: Can the GL-bit be used to turn unpredictability against a random sample and an adaptive challenge into indistinguishability against a random sample and an adaptive challenge?

The second part of Theorem 7 and the construction in [20] of full-fledged pseudo-random functions from *weak* pseudo-random functions give a relatively efficient transformation (compared with the transformation obtained by [12, 16, 17]) from the weakest notion considered in this paper (i.e. unpredictability against a random sample and a random challenge) to the stronger notion (i.e. indistinguishability against an adaptive sample and an adaptive challenge). An interesting task should be to achieve a more efficient transformation.

Section 5.1 considers the exact requirements for function families used in standard private-key schemes. An interesting line for further research discussed there is to design efficient private-key encryption and authentication schemes that only use *weak* pseudo-random functions. Implementations of such schemes may be very efficient given the extreme efficiency of candidates for weak pseudo-random functions.

Acknowledgments

We thank Ran Canetti and Benny Pinkas for motivating the problem. We thank Oded Goldreich for encouragement and many helpful comments. We also thank the reviewers of CRYPTO '98 and Hugo Krawczyk for many helpful comments.

References

1. M. Bellare, R. Canetti and H. Krawczyk, Keying hash functions for message authentication, *Proc. Advances in Cryptology - CRYPTO '96*, LNCS, Springer, vol. 1109, 1996, pp. 1-15.
2. M. Bellare, A. Desai, E. Jokipii and P. Rogaway, A Concrete Security Treatment of Symmetric Encryption, *Proc. 38th IEEE Symp. on Foundations of Computer Science*, 1997, pp. 394-403.
3. 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.
4. M. Bellare and S. Goldwasser, New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs *Proc. Advances in Cryptology - CRYPTO '89*, LNCS, Springer, 1990, pp. 194-211.
5. A. Blum, M. Furst, M. Kearns and R.J. Lipton, Cryptographic primitives based on hard learning problems, in: D.R. Stinson, ed., *Advances in Cryptology - CRYPTO '93*, LNCS, vol. 773, Springer, 1994, pp. 278-291.
6. M. Blum and S. Micali, How to generate cryptographically strong sequence of pseudo-random bits, *SIAM J. Comput.*, vol. 13, 1984, pp. 850-864.
7. G. Brassard, **Modern cryptology**, LNCS, vol. 325, Springer, 1988.
8. R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, Multicast security: A taxonomy and efficient authentication, manuscript.
9. D. Dolev, C. Dwork and M. Naor, Non-malleable cryptography, *Proc. 23rd Ann. ACM Symp. on Theory of Computing*, 1991, pp. 542-552. Full version available at: `http://www.wisdom.weizmann.ac.il/~naor`.

10. O. Goldreich, Two remarks concerning the Goldwasser-Micali-Rivest signature scheme, *Advances in Cryptology - CRYPTO' 86*, LNCS, vol. 263, 1987, pp. 104-110.
11. O. Goldreich, **Foundations of Cryptography (Fragments of a Book)**, 1995. Electronic publication in the Electronic Colloquium on Computational Complexity: `http://www.eccc.uni-trier.de/eccc/info/ECCC-Books/eccc-books.html`.
12. O. Goldreich, S. Goldwasser and S. Micali, How to construct random functions, *J. of the ACM.*, vol. 33, 1986, pp. 792-807.
13. O. Goldreich, S. Goldwasser and S. Micali, On the cryptographic applications of random functions, *Advances in Cryptology - CRYPTO '84*, LNCS, vol. 196, Springer, 1985, pp. 276-288.
14. O. Goldreich and L. Levin, A hard-core predicate for all one-way functions, in: *Proc. 21st Ann. ACM Symp. on Theory of Computing*, 1989, pp. 25-32.
15. 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.
16. J. Hastad, R. Impagliazzo, L. A. Levin and M. Luby, Construction of a pseudorandom generator from any one-way function, To appear in *SIAM J. Comput.* Preliminary versions by Impagliazzo et. al. in *21st STOC*, 1989 and Hastad in *22nd STOC*, 1990.
17. R. Impagliazzo and M. Luby, One-way functions are essential for complexity based cryptography, *Proc. 30th FOCS*, 1989, pp. 230-235.
18. M. Luby, **Pseudo-randomness and applications**, Princeton University Press, 1996.
19. M. Luby and C. Rackoff, How to construct pseudorandom permutations and pseudorandom functions, *SIAM J. Comput.*, vol. 17, 1988, pp. 373-386.
20. 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.
21. M. Naor and O. Reingold, On the construction of pseudo-random permutations: Luby-Rackoff revisited, To appear in: *J. of Cryptology.* Preliminary version in: *Proc. 29th Ann. ACM Symp. on Theory of Computing*, 1997. pp. 189-199.
22. M. Naor and O. Reingold, Number-Theoretic constructions of efficient pseudorandom functions, *Proc. 38th FOCS*, 1997, pp. 458-467.
23. B. Preneel and P. C. van Oorschot, On the security of two MAC algorithms, *Advances in Cryptology - EUROCRYPT '96*, LNCS, vol. 1070, 1996, pp. 19-32.
24. R. L. Rivest, Chaffing and winnowing: confidentiality without encryption, *MIT Lab for Computer Science*, `http://theory.lcs.mit.edu/~rivest/chaffing.txt`, March 18, 1998. To appear in: RSA CryptoBytes, Summer 1998.
25. 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.
26. A. Shamir, On the generation of cryptographically strong pseudo-random number sequences, *ACM Trans. Comput. Sys.*, vol 1, 1983, pp. 38-44.
27. 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.
28. A. C. Yao, Theory and applications of trapdoor functions, *Proc. 23rd IEEE Symp. on Foundations of Computer Science*, 1982, pp. 80-91.