On Constructing 1-1 One-Way Functions

Oded Goldreich* Leonid A Levin † Noam Nisan ‡

June 12, 1995

Abstract

We show how to construct length-preserving 1-1 one-way functions based on popular intractability assumptions (e.g., RSA, DLP). Such 1-1 functions should not be confused with (infinite) families of (finite) one-way permutations. What we want and obtain is a single (infinite) 1-1 one-way function.

*Department of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel. Email: oded@wisdom.weizmann.ac.il. Research was supported in part by grant No. 92-00226 from the United States – Israel Binational Science Foundation (BSF), Jerusalem, Israel.

†Computer Science Department, Boston University, Boston, USA. Email: lnd@bu-cs.bu.ac.il.

‡Institute for Computer Science, Hebrew University, Jerusalem, Israel. Email: noam@cs.huji.ac.il.

1 Introduction

Given any one-way permutation (i.e., a length preserving 1-1 one-way function), one can easily construct an efficient pseudorandom generator. The construction follows the scheme given by Blum and Micali [3], using the fact that every one-way function has a hard-core bit [5]. Specifically, assume that f is such a function and let b be a hard core-bit for it (i.e., starting with a function f' we define $f(x,r) \stackrel{\text{def}}{=} (f'(x), r)$ and $b(x,r)$ as the inner-product mod 2 of the strings x and r when viewed as binary vectors of length $|x| = |r|$). Then, the pseudorandom generator G, on input a seed s outputs the sequence $b(s), b(f(s)), b(f(f(s))), b(f^3(s)), ...$

Pseudorandom generators can be constructed also based on arbitrary one-way functions [8]; yet, the known construction is very complex and inefficient. In fact, it is of no practical value. The construction in [6], which uses arbitrary *regular* one-way functions is more attractive in these respects, yet it is far less attractive than the simple construction outlined above. A similar situation occurs with respect to the construction of digital signature schemes (cf., [10] vs [15]). In general, 1-1 one-way functions currently offer simpler and more practical constructions (of more complex primitives) than offered by general one-way functions.

These facts were our initial motivation for trying to construct length-preserving 1-1 one-way functions. Such functions should not be confused with what is commonly referred to (especially in the "Crypto Community") as "one-way permutations" and which are actually infinite sets of finite functions – see definitions below. What we want is a single infinite function, which is both length-preserving and 1-1 (and needless to say one-way). We show how to construct such 1-1 one-way functions based on popular intractability assumptions such as the intractability of DLP and inverting RSA.

Indeed, some (but not all) of the constructions which use length-preserving 1-1 one-way functions can be modified so that families of one-way permutations can be used instead. Still the question of whether the former exists is of both theoretical and practical importance.

2 One-Way Functions and Families

Definition 1 (one-way functions): *Let* $f:\{0,1\}^* \mapsto \{0,1\}^*$ *be a* length preserving *function which is polynomial-time computable.*

- (strongly one-way): *f is called* (strongly) one-way *if for any probabilistic polynomial-time algorithm A, any positive polynomial p and all sufficiently large n's*

$$\text{Prob}(A(f(x)) \in f^{-1}f(x)) < \frac{1}{p(n)}$$

where the probability is taken uniformly over $x \in \{0,1\}^n$, and the internal coin tosses of algorithm A.

- (weakly one-way): *f is called* weakly one-way *if there exists a positive polynomial p so that for any probabilistic polynomial-time algorithm A and all sufficiently large n's*

$$\text{Prob}(A(f(x)) \notin f^{-1}f(x)) > \frac{1}{p(n)}$$

where the probability is as above.

(Note that $f:\{0,1\}^* \mapsto \{0,1\}^*$ is 1-1 if $f(x) \neq f(y)$ for all $x \neq y$). In case $f(x) \neq f(y)$ for all but a negligible fraction of the pairs (x,y) we say that f is **almost 1-1**. Namely, an almost 1-1 function f satisfies, for every positive polynomial p and all sufficiently large n's,

$$\mathrm{Prob}(f(x)=f(y)) < \frac{1}{p(n)}$$

where the probability is taken uniformly and indepedently over all $x, y \in \{0,1\}^n$.

Definition 2 (family of one-way permutations – simplified version): *A set of finite permutations,* $\mathbf{F} = \{f_i : D_i \overset{1\text{-}1}{\mapsto} D_i\}_{i \in I}$, *is called a* **family of one-way permutations** *if the following conditions hold*

- (efficient evaluation): *there exists a polynomial-time algorithm that on input an index* (of a permutation) $i \in I$ *and a domain element* $x \in D_i$ *returns* $f_i(x)$*.*
- (efficient index selection): *there exists a probabilistic algorithm* S *that on input* n*, runs for* $\mathrm{poly}(n)$ *time and returns a uniformly distributed index of length* n (i.e., an i uniformly distributed in $I \cap \{0,1\}^n$).
- (efficient domain sampling): *there exists a probabilistic polynomial-time algorithm* D *that on input an index* $i \in I$*, returns a uniformly distributed element of* D_i*.*
- (one-wayness): *For any probabilistic polynomial-time algorithm* A*, any positive polynomial* p *and all sufficiently large* n*'s*

  $$\mathrm{Prob}(A(i, f_i(x)) = x) < \frac{1}{p(n)}$$

  where the probability is taken uniformly over $i \in I \cap \{0,1\}^n$*,* $x \in D_i$*, and the internal coin tosses of algorithm* A*.*

In the **non-simplified version**, both probabilistic algorithms mentioned above (i.e., S and D) are allowed to produce output with only non-negligible probability (i.e., probability at least $1/\mathrm{poly}(n)$). Furthermore, given these algorithms have produced an output, the output is allowed to be wrong (i.e., out of the target set or non-uniformly distributed) with negligible probability (e.g., with probability at most 2^{-n}).

Analogously to Definition 1, families of permutations can be defined to be *weakly* one-way, rather than (strongly) one-way.

3 Transforming One-Way Families into Functions

Clearly, any family of one-way permutations can be converted into a single one-way function; namely, $f(r,s) \stackrel{\mathrm{def}}{=} f_i(x)$, where $i = S(n,r)$ is the index selected using coin-tosses r and $x = D(i,s) \in D_i$ is the element selected on input i and coin-tosses s. (Padding can be applied, if necessary, to make f length preserving.) However, this procedure does not necessarily yield a 1-1 function; furthermore, for most natural examples such as RSA, DLP, etc., the resulting function will be many-to-one.

An alternative construction, which does yield a 1-1 one-way function, is possible under some additional conditions, as demonstrated below. In fact, the conditions are defined to make this natural construction work and the thrust of this paper is in demonstrating that these conditions can be met under reasonable and popular assumptions (see next section).

3.1 The Conditions

Let **F** be a family of one-way permutations and that let $q(n)$ denote the number of coins flipped by the index selection algorithm S on input n. We consider the following conditions that **F** may satisfy

Definition 3 (additional conditions)

- augmented one-wayness: *For any probabilistic polynomial-time algorithm A, any positive polynomial p and all sufficiently large n's*

$$\mathrm{Prob}(A(r, f_{S(n,r)}(x)) = x) < \frac{1}{p(n)}$$

  where the probability is taken uniformly over $r \in \{0,1\}^{q(n)}$, $x \in D_{S(n,r)}$, and the internal coin tosses of algorithm A.
  (Namely, the permutations are hard to invert even when the inverting algorithm is given the random coins used to generate the index of the permutation.)

- canonical domain sampling: *The domain-sampling algorithm may consist of uniformly selecting a string of specific* (easy to determine) *length and testing whether the string resides in the domain. In other words, we require*
  - (recognizable domain): *There exists a polynomial-time algorithm that on input an index $i \in I$ and a string x decides if $x \in D_i$.*
  - (non-negligible domain): *There exists a polynomial-time computable function $l : \mathbb{N} \mapsto \mathbb{N}$ and a positive polynomial $p(\cdot)$ so that $D_i \subseteq \{0,1\}^{l(n)}$ and $|D_i| > \frac{1}{p(n)} \cdot 2^{l(n)}$*

3.2 The Construction

Given a family of one-way permutations that satisfies the additional conditions, we explicitly construct a 1-1 one-way function as follows.

Construction 1 (simple version): *Let **F** be a family of permutations with an index selection algorithm S that uses $q(\cdot)$ coins and having domains D_i's which are subsets of $\{0,1\}^{l(|i|)}$, for some function $l(\cdot)$. We construct the function f as follows*

$$f(r,s) \stackrel{\mathrm{def}}{=} \begin{cases} (r, f_i(s)) & \text{if } s \in D_i, \text{ where } i \stackrel{\mathrm{def}}{=} S(n,r) \\ (r,s) & \text{otherwise} \end{cases}$$

where $r \in \{0,1\}^{q(n)}$ and $s \in \{0,1\}^{l(n)}$,

Proposition 1 : *The function f is 1-1 and length preserving. If* $\mathbf{F}$ *is a family of one-way permutations satisfying the additional conditions of Definition 3 then f is weakly one-way. The latter holds even if* $\mathbf{F}$ *is only weakly one-way* (and satisfies the additional conditions).

proof: By definition f is length-preserving. Let G_n be the set of pairs $(r,s) \in \{0,1\}^{q(n)} \times \{0,1\}^{l(n)}$ so that $s \in D_{S(n,r)}$ holds and let B_n be the set of the other pairs (i.e., $B_n = (\{0,1\}^{q(n)} \times \{0,1\}^{l(n)}) - G_n$). The key observation is that if $(r,s) \in G_n$ then, letting $i = S(n,r)$, $s \in D_i$ holds and $f_i(s) \in D_i$ (and $f(r,s) \in G_n$) follows. On the other hand, if $(r,s) \in B_n$ then $f(r,s) = (r,s) \in B_n$. Thus, f maps G_n (resp., B_n) to itself and furthermore the mapping induced on G_n (rep., B_n) is 1-1. It follows that f is 1-1.

The function f is polynimal-time computable by virtue of the first two efficiency conditions of $\mathbf{F}$ and the additional 'recognizable domain' condition. By the additional 'non-negligible domain' condition we know that G_n forms a non-negligible fraction of $G_n \cup B_n$ and by the 'augmented one-wayness' condition we infer that f is hard to invert on G_n. Thus, we conclude that f is weakly one-way. In fact, the latter conclusion remain valid even if the family of permutations $\mathbf{F}$ is only weakly one-way. □

Remark: The function f (constructed above) may be only weakly one-way since it equals the identity transformation for a part of its domain and this part may have non-negligible measure. To get a (strongly) one-way function, one may apply the transformation in [4] to the function f. (In fact, degenerate versions of the transformation in [4] suffice for this purpose.)

The above construction is stated with respect to the simplified definition of a family of one-way permutations. Recall that in the non-simplified version, the index-selecting algorithm, S, is only required to have an output with non-negligible probability (i.e., the probability is bounded below by $1/p(n)$ where p is some fixed positive polynomial). Furthermore, S is allowed to err (i.e., have output not in I) with a negligible probability. For the general case, we redefine the function f as follows

Construction 2 (complex version): *Let* $\mathbf{F} = \{f_i : D_i \overset{1\text{-}1}{\mapsto} D_i\}_{i \in I}$ *be a family of permutations with an index-selecting algorithm, S, which produces output with non-negligible probability and errs with negligible probability. We construct the function f as follows*

$$f(r,s) \stackrel{\text{def}}{=} \begin{cases} (r, f_i(s)) & \textit{if } i \stackrel{\text{def}}{=} S(n,r) \neq \perp \textit{ and } s \in D_i \\ (r,s) & \textit{otherwise} \end{cases}$$

where the convention is that in case, on input n and coin tosses $r \in \{0,1\}^{q(n)}$, the algorithm S halts with no output then $S(n,r) \stackrel{\text{def}}{=} \perp \notin \{0,1\}^*$.

Proposition 2 : *The function f is length preserving and almost 1-1, and in case S never errs f is 1-1. If* $\mathbf{F}$ *is a family of one-way permutations satisfying the additional conditions of Definition 3 then f is weakly one-way. The latter holds even if* $\mathbf{F}$ *is only weakly one-way* (and satisfies the additional conditions).

proof: In case algorithm S never errs, the proof is similar to the proof of the previous proposition (i.e., G_n is defined as the set of all pairs (r,s) so that $i \stackrel{\text{def}}{=} S(n,r) \neq \perp$ and $s \in D_i$). Otherwise, we observe that the collision probability of f is bouded above by the probability that S errs (and outputs a string not in I). Since this happens with negligible probability, we are done. □

4 Applying the Transformation

Using the transformation specified in the previous section, we show how to construct a 1-1 one-way function based on one of several popular intractability assumptions. To this end, we use these intractability assumptions in order to construct families of one-way permutations satisfying the additional conditions of Definition 3. Before presenting these constructions, we wish to stress an important aspect regarding them; namely, their "security".

Security

The security of a one-way function f is a function, $s : \mathbb{N} \mapsto \mathbb{N}$, specifying the amount of "work" required to invert f on inputs of given length. The work of an algorithm is defined as the product of the running-time (of the inverting algorithm) and the inverse of its success probability; namely, $w_A(n) \stackrel{\rm def}{=} t_A(n) \cdot \frac{1}{p_A(n)}$, where $t_A(n)$ is the running time of A on f-images of length n and $p_A(n) \stackrel{\rm def}{=} {\rm Prob}_{x \in \{0,1\}^n}(A(f(x)) \in f^{-1}f(x))$ is its success probability.

Typical cryptographic constructions, and in particular our constructions, transform one object (in our case a family of one-way permutations) of security $s(\cdot)$ into another object (in our case a single 1-1 one-way function) of related security $s'(\cdot)$. The relation between s and s' is of key importance. A weak relation, which is usually easier to obtain, is that $s'({\rm poly}(n)) > s(n)/{\rm poly}(n)$. Although this relation translates any super-polynomial security s into a superpolynomial security s', it is of limited practical value. In order to use the resulting object of security s' one may needs to use very big instances. For example, if $s'(n^5) = s(n)$ and the original object is "secure in reality" for instance size 100 (bits) then the resulting object (of security s') will be "secure in reality" only for instances of size 10^{10}, and is thus unlikely to be of practical value. Thus, stronger relation between the security s of the original object and the security s' of the resulting object are of more value. In particular, it is desirable to have $s'(O(n)) > s(n)/{\rm poly}(n)$, in which case we say that the transformation preserves the security.

Getting back to the constructions of the previous section, we note that the security of the resulting one-way 1-1 function f, on f-images of length $q(n)+l(n)$, is at least a polynomial fraction of the security of the family of one-way permutations on f_i-images of length $l(n)$. (Recall, n denotes the length of the index of the permutation, $l(n)$ the length of the description of elements in the domain of the permutation and $q(n)$ the randomness complexity of the index-selecting algorithm.) Namely, $s'(q(n)+l(n)) > s(l(n))/{\rm poly}(n)$, where s denotes the security of the family $\mathbf{F}$ and s' the security of the function f. Therefore, the smaller the polynomial $q(\cdot)$ is, the better security one gets. It is particularly desirable to keep $q(n)$ linear in $l(n)$. *All the constructions presented below achieve this goal. Consequently, the one-way functions constructed below preserve the security of the intractability assumption on which they are based.* We remark that the (weak to strong one-way) transformation of [4] (mentioned in the Remark above) preserves security too.

Preliminaries: selecting prime numbers

Prime numbers play a key role in all our constructions and so efficient algorithms for selecting such numbers are of key importance to us. We will use two algorithms due to Bach [1, 2]. The first algorithm [2] is merely a very efficient (problem-specific) "deterministic amplification" of the

Miller-Rabin primality tester [9, 13]. The second algorithm [1] produces uniformly distributed integers together with their prime factorization.

Theorem 1 (randonmess efficient primality tester [2]): *There exists a probabilistic polynomial time algorithm that on input P uses $|P|$ random bits so that if P is prime then the algorithm always accepts, and otherwise* (i.e., P is composite) *the algorithm accepts with probability at most* $\frac{1}{\sqrt{P}} = 2^{-|P|/2}$.

Theorem 2 (space efficient generation of integers with known prime factorization [1]): *There exists a probabilistic polynomial time algorithm that uses linear space and on input* 1^n *uniformly generates a number N in the interval* $[2^{n-1}, 2^n - 1]$ *and outputs the prime factorization of N.*

The above two algorithms are reasonablly efficient. We are reluctant to use the primality certifier of Goldwasser and Kilian which for all but a negligible fraction of the primes finds in probabilistic polynomial-time a certificate of primality [7]. Interestingly, this algorithm can be implemented within linear space and so applying the transforamtion of Nisan and Zuckerman [11] we get an implementation which uses linear randomness.

Theorem 3 (randonmess efficient primality certifier [7, 11]): *There exists a probabilistic polynomial time algorithm that on input P uses $O(|P|)$ random bits and for all but a negligible fraction of the primes finds a certificate of primality* (i.e., a witness/proof with respect to some NP-relation).

4.1 A construction based on RSA

The standard presentation of RSA [14] yields a family of permutations which is believed to be one-way, but is certainly **not** one-way in the augmented sense of Definition 3. (Here we refer to a family in which the indices are pairs (N, e), where N is the product of two primes of equal length and e is relatively prime to $\phi(N)$. The index is generated by randomly selecting these two primes, multiplying them and next selecting a proper e. Thus, giving these random choices away compromises the security of RSA, since given the prime factors it is easy to invert the function.)

We consider, instead, the following family of weak one-way permutations. The indices in this family are pairs of integers (N, P) so that P is a prime and $|P| = |N|$. For each such pair we define a permutation over $\mathbf{Z}_N^*$, the multiplicative group modulo N; specifically, $f_{N,P}(x) \stackrel{\text{def}}{=} x^P \bmod N$. Note that we do not insist that N is a product of two primes of the same length. Yet, a non-negligible fraction of the possible N's will have this form. Thus, if the standard RSA family is strongly one-way (for random exponent) then it is also (strongly) one-way for a prime exponent and consequently the above (non-standard) family of functions will be weakly one-way (due to the non-negligible fraction of composites of the standard form). Since P is relatively-prime to $\phi(N)$, the functions in this family are in fact permutations over $\mathbf{Z}_N^*$. (Note that the index-selecting algorithm does not know $\phi(N)$ and so relative-primality of P and $\phi(N)$ is imposed by requiring that P is prime.)

We now show that the above family satisfies the non-simplified requirements (from a family of one-way permutations) as well as the additional conditions in Definition 3. Of the efficiency conditions only the index selection is problematic, yet it does hold when allowing negligible error and requiring that output is produced only with non-negligible probability (i.e., just select two n-bit integers at random and check if the second is prime – producing an output only if the answer is in the affirmative). Also, $\mathbf{Z}_N^*$ is easily recognizable and is non-negligible with respect to $\{0,1\}^{|N|}$. Furthermore, this family is one-way in the augmented sense (under the "RSA assumption") since the modulus is generated via an identity transformation from the coins of the index-selecting algorithm (and thus these coins add no knowledge to the inverter). It follows that we can apply Proposition 2 and derive an almost 1-1 one-way function.

Definition 4 (standard RSA Assumption): *We say that* inverting RSA is intractable with security $s(\cdot)$ *if any algorithm for the* inverting task *uses work greater than* $s(\cdot)$. *The inverting task consists of finding* x *such that* $y = x^e \bmod N$, *when given* N, e *and* y, *where* N *is uniformly selected among all composites which are the product of two* $(n/2)$*-bit long primes,* e *is uniformly selected among the elements of the multiplicative group modulo* $\phi(N)$, *and* y *is uniformly selected among the elements of the multiplicative group modulo* N.

To justify our claim that security (of the RSA Assumption) is preserved we have to note that pairs (N, P) as required can be selected using $O(|(N, P)|)$ random bits. To this end, we use the algorithm guaranteed in Theorem 1. Thus, we get

Corollary 1 : *Suppose that inverting RSA is intractable with security* $s(n)$. *Then, there exists an almost 1-1 one-way function with security* $s'(O(n)) \stackrel{\rm def}{=} s(n)/\mathrm{poly}(n)$.

The possible colisions in the one-way function are due to the error probability of the index selection algorithm which in turn is due to the probability that a composite passes the primality test. Using Theorem 3 we can get rid of this error (i.e., if we fail to generate a certificate then we treat the integer, which is possibly a prime, as if it were found to be composite). Thus, assuming that inverting RSA is intractable (with security $s(n)$), there exists a 1-1 one-way function (with security $s'(O(n)) \stackrel{\rm def}{=} s(n)/\mathrm{poly}(n)$).

4.2 A construction based on a restricted DLP

Here we rely on the assumption that the Discrete Logarithm Problem (DLP) in the multiplicative group modulo P is hard also for the special case of primes of the form $P = 2Q + 1$, where Q is a prime. We also use the assumption that such primes form a non-negligible fraction of the integers of the same length. Based on these assumptions, the following family of permutations is one-way. The indices in the family are pairs (P, g), where P is a prime of the above form and g is a primitive element modulo P. The index is selected by first selecting a prime of the above form and next using the known factorization of $\phi(P) = 2Q$ to test candidates for primitivity (see details below). For each index, (P, g), we define a permutation over $\mathbf{Z}_P^*$, the multiplicative group modulo P; specifically, $f_{P,g}(x) \stackrel{\rm def}{=} g^x \bmod P$. Noting that $\mathbf{Z}_P^*$ is both 'non-negligible' and easy to recognize, we can apply Proposition 2.

To justify our claim that the resulting 1-1 one-way function preserves the security of the family, we note that pairs (P, g) can be selected using $O(|P|)$ random bits. On input n we uniformly select an $(n-1)$-bit integer, Q, and test Q and $P = 2Q + 1$ for primality. In case we are successful, we uniformly select $g \in \mathbf{Z}_P^*$ and test if it is primitive (mod P) by computing $g^{P-1} \bmod P$, $g^Q \bmod P$ and $g^2 \bmod P$. (If the first expression evaluates to 1 whereas the other two don't, then g is a primitive element modulo P.) We get

Corollary 2 : *Suppose that the restricted DLP is intractable with security* $s(n)$ (see definition below)*, and that the set of* n*-bit primes,* P*, for which* $\phi(P)/2$ *is prime, constitute a* $1/\text{poly}(n)$ *fraction of the* n*-bit long integers. Then, there exists an almost 1-1 one-way function with security* $s'(O(n)) \stackrel{\text{def}}{=} s(n)/\text{poly}(n)$.

Again, the one-way function can be made 1-1 by using Theorem 3 (as above).

Definition 5 (restricted DLP Assumption): *We say that* the restricted DLP is intractable with security $s(\cdot)$ *if any algorithm for the following* inverting task *uses work greater than* $s(\cdot)$*. The inverting task consists of finding* x *such that* $y = g^x \bmod P$*, when given* P*,* g *and* y*, where* P *is uniformly selected among all* n*-bit primes for which* $\phi(P)/2$ *is prime,* g *is uniformly selected among the primitive elements modulo* P*, and* y *is uniformly selected among the elements of the multiplicative group modulo* P*.*

4.3 A construction based on the general DLP

Here we rely on a seemingly weaker assumption concerning the DLP. Specifically, we assume that the Discrete Logarithm Problem (DLP) in the multiplicative group modulo a prime P is hard also when given the factorization of $\phi(P)$. Making this assumption, we can waive the assumption made in the previous subsection concerning the density of primes of special form $P = 2Q + 1$, where Q is a prime. (Note that for primes of the special form $P = 2Q + 1$ the factorization of $\phi(P) = 2 \cdot Q$ is always known.)

Based on the above intractability assumption, the following family of permutations is one-way. The indices in the family are pairs (P, g), where P is a prime and g is a primitive element modulo P. The index is chosen by first generating a random prime P with known factorization of $\phi(P)$ (see details below), and next using this factorization to test candidates for primitivity. For each index, (P, g), we define a permutation over $\mathbf{Z}_P^*$ as before (i.e., $f_{P,g}(x) \stackrel{\text{def}}{=} g^x \bmod P$). Again, we can apply Proposition 2.

We have postponed the discussion of how to randomly generate primes P with known factorization of $\phi(P)$. Here, a different algorithm of Bach comes to the rescue. This algorithm, uniformly generates composites with their factorization [1]. Having produced a factored composite N, we test $N + 1$ for primality and are done if the answer is in the affirmative. Actually, the algorithm produces a certificate for the primality of $P = N + 1$ in probabilistic polynomail-time using the (certified) factorization of $P - 1$ (produced by Bach's algorithm). A straightforward implementation of Bach's algorithm requires a super-linear number of coin tosses. Yet, it is possible to implement an approximation of the algorithm using only a linear number of coin tosses (i.e., linear in the length of the composite being generated). The details are quite tedious. Instead, we prefer to invoke a general result of Nisan and Zuckerman [11] by which any probabilistic polynomial-time algorithm, which uses linear space, can be approximated using a linear number of coin tosses. It is very easy to see that Bach's algorithm falls into this category (and this is stated in Theorem 2 above). This yields an index selecting algorithm which selects pairs (P, g) using $O(|(P, g)|)$ random bits, justifying our claim that the resulting 1-1 one-way function preserves the security of the family. We stress that the index selecting algorithm never errs (and furthermore it produces a certificante for membership in the index set). Thus, we get

Corollary 3 : *Suppose that DLP is intractable with security $s(n)$, even when the factorization of the order of the group is given* (see definition below). *Then, there exists a 1-1 one-way function with security* $s'(O(n)) \stackrel{\text{def}}{=} s(n)/\text{poly}(n)$.

Definition 6 (DLP Assumption): *We say that* the DLP is intractable with security $s(\cdot)$ *if any algorithm for the following* inverting task *uses work greater than $s(\cdot)$. The inverting task consists of finding x such that $y = g^x$* mod *P, when given P, the factorization of $\phi(P)$, g and y, where P is uniformly selected among all n-bit primes, g is uniformly selected among the primitive elements modulo P, and y is uniformly selected among the elements of the multiplicative group modulo P.*

5 Conclusions and Open Problems

We have presented a method for constructing (strongly) one-way permutations. The method consists of three steps.

Step (1) using well-known intractability assumptions to construct families of one-way permutations satisfying the additional properties specified in Definition 3;

Step (2) using such a family to construct a weak one-way function;

Step (3) transforming the resulting function into a strongly one-way function.

We consider the identification of the conditions in Definition 3 and the construction of families of one-way permutations satisfying these conditions to be the most important contributions of the current paper. Thus, most of the paper is dedicated to the implementation of Step (1), whereas Step (2) is obtained by Construction 1 and Step (3) is obtained by referring to [4].

Regarding Step (3), we remark that applying the general ("weak to strong") transformation of [4] seems an over-kill since in our case the weakly one-way function f has a special structure (e.g., it is hard to invert almost on all points on which it is not the identity transformation). Furthermore, it seems that ad-hoc methods may be applicable to the function f resulting from a specific transformation. However, in our attempts to avoid using [4], we were not able to avoid using random walks on expander graphs (and since expander graphs are the only non-elementary component of [4] we see no point in presenting these alternatives here). Certainly, it will be better to the use of expander graphs and perform Step (3) in a more efficient manner.

Another obvious open problem is to construct one-way 1-1 functions based on the intractability of factoring. To achieve this goal using our method one will need to construct a family of one-way permutations satisfying the additional properties specified in Definition 3. (The standard construction of a family of one-way permutations based on factoring [12] does not satisfy the augmented one-wayness condition.)

Acknowledgments

We would like to thank Eric Bach and Hugo Krawczyk for helpful discussions and comments.

References

[1] E. Bach, Analytic Methods in the Analysis and Design of Number-Theoretic Algorithms (ACM Distinguished Dissertation 1984), MIT Press, Cambridge MA, 1985.

[2] E. Bach, "Realistic Analysis of some Randomized Algorithms", 19th STOC, 1987, pp. 453–461.

[3] M. Blum and S. Micali, "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits", SIAM J. on Computing, Vol. 13, 1984, pp. 850–864.

[4] O. Goldreich, R. Impagliazzo, L. Levin, R. Venkatesan and D. Zuckerman, "Security Preserving Amplification of Hardness", 31st FOCS, 1990, pp. 318–326.

[5] O. Goldreich and L. Levin, "A Hard-Core Predicate for any One-way Function", 21st STOC, 1989, pp. 25–32.

[6] O. Goldreich, H. Krawczyk and M. Luby, "On the Existence of Pseudorandom Generators", SIAM J. on Computing, Vol. 22, 1993, pp. 1163–1175.

[7] S. Goldwasser and J. Kilian, "Almost all primes can be quickly certified", 18th STOC, 1986, pp. 316–329.

[8] J. Håstad, R. Impagliazzo, L. Levin and M. Luby, "Construction of a pseudo-random generator from any one-way function", *ICSI Technical Report*, No. 91-068, submitted to *SICOMP*. Combines papers of Impagliazzo, Levin and Luby (21st STOC, 1989, pp. 12–24) and J. Håstad, (22nd STOC, 1990, pp. 395–404).

[9] G.L. Miller, "Riemann's Hypothesis and tests for primality", JCSS, Vol. 13, pp. 300–317, 1976.

[10] M. Naor and M. Yung, "Universal Hash Functions and their Cryptographic Applications", 21st STOC, 1989, pp. 33–43.

[11] N. Nisan and D. Zuckerman, "More Deterministic Simulation in LOGSPACE", 25th STOC, 1993, pp. 235–244.

[12] M.O. Rabin, "Digitalized Signatures and Public Key Functions as Intractable as Factoring", MIT/LCS/TR-212, 1979.

[13] M.O. Rabin, "Probabilistic algorithm for testing primality", Jour. of Number Theory, Vol. 12, pp. 128–138, 1980.

[14] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", CACM, Vol. 21, Feb. 1978, pp. 120–126.

[15] J. Rompel, "One-way Functions are Necessary and Sufficient for Secure Signatures", 22nd STOC, 1990, pp. 387–394.

[16] R. Solovay and V. Strassen, "A fast Monte-Carlo test for primality", SIAM Jour. on Computing, Vol. 6, pp. 84–85, 1977.