Limits on the Efficiency of One-Way Permutation-Based Hash Functions

Jeong Han Kim
Daniel R. Simon
Microsoft Research
One Microsoft Way
Redmond WA 98052 USA
email: jehkim@microsoft.com,
dansimon@microsoft.com

Prasad Tetali
School of Mathematics
Georgia Tech
Atlanta, GA 30332-0160
email: tetali@math.gatech.edu

February 3, 1999

Technical Report
MSR-TR-99-06

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Limits on the Efficiency of One-Way Permutation-Based Hash Functions

Jeong Han Kim
Daniel R. Simon
Microsoft Research
One Microsoft Way
Redmond WA 98052 USA
email: jehkim@microsoft.com,
dansimon@microsoft.com

Prasad Tetali
School of Mathematics
Georgia Tech
Atlanta, GA 30332-0160
email: tetali@math.gatech.edu

February 3, 1999

Abstract

Naor and Yung show that a one-bit-compressing universal one-way hash function (UOWHF) can be constructed based on a one-way permutation. This construction can be iterated to build a UOWHF which compresses by εn bits, at the cost of εn invocations of the one-way permutation. We show that this construction is not far from optimal, in the following sense: there exists an oracle relative to which there exists a one-way permutation with inversion probability $2^{-p(n)}$ (for any $p(n) \in \omega(\log n)$), but any construction of an n-bit-compressing UOWHF requires $\Omega(\sqrt{n/p(n)})$ invocations of the one-way permutation, on average. (For example, there exists in this relativized world a one-way permutation with inversion probability $n^{-\omega(1)}$, but no UOWHF that invokes it fewer than $\Omega(\sqrt{n}/\log n)$ times.) Thus any proof that a more efficient UOWHF can be derived from a one-way permutation is necessarily non-relativizing; in particular, no provable construction of a more efficient UOWHF can exist based solely on a "black box" one-way permutation. This result can be viewed as a partial justification for the practice of building efficient UOWHFs from stronger primitives (such as collision-intractable hash functions), rather than from weaker primitives such as one-way permutations.

Key words: Oracle, relativization, cryptography, complexity theory

1 Introduction

A universal one-way hash function (UOWHF) is a family of length-decreasing functions such that for any input x, it is computationally infeasible to find a collision with x (a second input giving the same output) under a function chosen randomly from the UOWHF family. UOWHFs were introduced by Naor and Yung ([NY89]), who proved that they can be constructed given any one-way permutation, and that moreover they suffice for constructing a number of cryptographic tools, including digital signature schemes. Later Rompel ([Rom90]) showed how to construct a UOWHF from any one-way function (not necessarily a permutation). One drawback of the constructions in [NY89] and [Rom90] is their inefficiency; they require at least one invocation of a one-way permutation for every bit of length decrease effected by the hash function. (This efficiency can easily be improved to one invocation per $\log n$ bits of length decrease, but it is not obvious how to improve it further.) As a result, UOWHFs based on one-way functions or one-way permutations (or their equivalents, such as block ciphers) are not widely used in practice; instead, collision-intractable hash functions such as MD5 ([Riv92]) and SHA-1 ([NIST94]) are typically used.

It is natural to ask if more efficient provable constructions of UOWHFs based on one-way permutations are possible. Here, we answer that question in the negative, showing that the construction of [NY89] is not far from optimal, in the following sense: there exists an oracle relative to which there exists a one-way permutation with inversion probability $2^{-p(n)}$, but any construction of an εn-bit-compressing UOWHF (that is, one that maps n-bit inputs to $(1-\varepsilon)n$-bit outputs, for some constant ε) requires $\Omega(\sqrt{n/p(n)})$ invocations of the one-way permutation. In particular, a one-way permutation whose inversion probability is only known to be $n^{-\omega(1)}$ can only be used to construct a UOWHF relative to this oracle if the UOWHF invokes the one-way permutation at least $\Omega(\sqrt{n}/\log n)$ times on average. Thus any proof that a more efficient UOWHF can be derived from a one-way permutation is necessarily non-relativizing; in particular, no provable construction of a more efficient UOWHF can exist based solely on a "black box" one-way permutation. This result can be viewed as a partial justification for the practice of building efficient UOWHFs from stronger primitives (such as collision-intractable hash functions—see [BR97]), rather than from weaker primitives such as one-way permutations (as was first proposed in [NY89]).

The method used in the proof is similar to that of [Sim98]; a random permutation oracle is used as the one-way permutation, and is shown to be one-way even in the presence of a collision-finding oracle. In this case, however, the collision-finding oracle must be much weaker, since an oracle that finds collisions in all one-way-permutation-based UOWHFs—including known, provable constructions—would hence necessarily be able to invert the underlying one-way permutation. Instead, a combinatorial argument is used to show that a particular weak collision-finding oracle can find collisions in any UOWHF that makes insufficient use of a one-way permutation.

2 Definitions

We review here some basic definitions.

Definition 1 *A* $q(n)$-one-way permutation f *is a family* $\{f_n\}$ *of polynomial-time computable permutations on* n*-bit strings such that for any non-uniform polynomial-size circuit family* $C = \{C_n\}$ *the probability that* C_n *outputs* x *on input* $f_n(x)$ *for a uniformly chosen* $x \in \{0,1\}^n$ *is at most* $q(n)$*. (For any* n*-bit* x*,* $f(x)$ *is used to denote* $f_n(x)$*.)*

Definition 2 *([NY89])A* universal one-way hash function family *is a family* $H = \{H_{h,n} : \{0,1\}^n \to \{0,1\}^{m(n)}, m(n) < n\}$ *of polynomial-time computable functions such that for any non-uniform polynomial-size circuit family* $C =$

$\{C_n\}$, *the probability is* $n^{-\omega(1)}$ *that* C_n, *given input* (h, x) *with* $h \in \{0,1\}^{p(n)}$ *(p a polynomial) and* x *uniformly chosen from their domains, outputs a* $y \neq x$ *such that* $H_{h,n}(y) = H_{h,n}(x)$.

3 The Main Result

The intuition underlying the theorem and proof is as follows: consider a generic construction of a UOWHF $H = \{H_{h,n}\}$ from a (black-box) one-way permutation f that compresses an n-bit input to $(1-\varepsilon)n$ bits (for example, an iterated version of the one presented in [NY89]). The "colliding set" $S(x)$ of inputs colliding with a given n-bit x under $H_{h,n}$ is therefore of size $2^{\varepsilon n}$ (on average). Suppose that f is invertible (by some inverting algorithm) on some tiny fraction of its inputs—say, $2^{-p(n)}$ (where $p(n) \in \omega(\log n)$). Since f can be an arbitrary one-way permutation, we can assume that the set R of invertible inputs to f is not chosen optimally for the security of H; let us say, then, that this "chosen set" R is generated by selecting inputs uniformly at random. Moreover, since we make no *a priori* assumptions about the collision-intractability or invertibility of H, we must assume that if an element s of colliding set $S(x)$, when input into $H_{h,n}$, causes f to be computed only on elements of chosen set R, then an adversary can reverse the computation of $H_{h,n}(s)$ (since each of its invocations of f is invertible), and recover s. Hence if such a member of $S(x)$ exists, then there is no assurance that the adversary cannot find it, invert it, and thus find a collision with x.

Recasting this intuition in relativized terms, consider an oracle A which accepts queries in the form of either n-bit strings or pairs containing an n-bit string and a description of a circuit with n-bit-input oracle gates. Given an n-bit string x, A will compute $f(x)$ for some fixed random permutation f. Given a string-circuit description pair (x, C), it will treat the oracle gates in the circuit as queries to compute f (i.e., as "f-queries") and select a random "chosen set" R of $\{0,1\}^n$ (independently for each distinct pair (x, C)) of size roughly $2^{n-p(n)}$ (where $p(n) \in \omega(\log n)$). A will then return an s (if it exists) such that (1) $C(x) = C(s)$ (i.e., $s \in S(x)$), and (2) every f-query in the computation of $C(s)$ has an input which is in R. (The randomness can ultimately be removed from the oracle, of course, but the argument is simpler if A is assumed to be randomized.)

This oracle will never invert a random f on a random input with better than $2^{-O(p(n))}$ probability, as long as it's only queried polynomially many times, since it only ever reveals any information about a random fraction $2^{-O(p(n))}$ of the input-output pairs. Hence f remains a one-way permutation even in the presence of this oracle. The question is whether there exists a choice of C—that is to say, a UOWHF construction—which minimizes the probability that A will find an s. Equivalently, C must minimize the probability that the randomly chosen set R will "cover" (that is, include all the f-queries in the computation of C when the input is) at least one member of colliding set $S(x)$. We will call this probability the *cover probability* of x under C.

For example, suppose that the inputs to the f-queries in C are distinct—or at least independently distributed—over $\{0,1\}^n$ for each distinct input to C. Then a simple probability calculation shows that the cover probability of a randomly chosen x under C is high unless there are $\Omega(n/p(n))$ f-queries, on average, in the computation of C on an input in $S(x)$. (After all, there are on average $2^{\varepsilon n}$ members of $S(x)$; in order for none of them to be covered by R, the probability that each one is covered must be very small—meaning that there must be many f-queries for each member.)

Of course C need not be constructed so that f-queries are independently distributed for different circuit inputs. For example, if the same f-queries are used for all the circuit inputs in $S(x)$, then the probability that they are all covered is $n^{-\omega(1)}$ unless that set of common f-queries is of size at least $n^{\omega(1)}$. On the other hand, there is another way to find the inverse of some f-queries: by using the fact that x is known, and hence any f-queries that are used in the computation of $C(x)$ are also known to an adversary trying to find a collision with x. We will therefore modify A so that these invertible input-output pairs (which make up a set we will denote by $Q(x)$) are added to the chosen set R, along with the randomly chosen ones. (Note that adding these values—which can be computed anyway for a random x in polynomial time—doesn't alter the one-wayness of f.)

Thus the modified A, given an n-bit string x, will compute $f(x)$ for some fixed random permutation f, and given a string-circuit description pair (x, C), it will treat the oracle gates in the circuit as queries to compute f ("f-queries"), select a chosen set $R \subset \{0,1\}^n$, of size roughly $2^{n-p(n)}$, uniformly at random, and return an s (if it exists) such that (1) $s \in S(x)$, and (2) every f-query in the computation of $C(s)$ has an input which is in the set $R' = R \cup Q(x)$. We will prove that any choice of C will with non-negligible probability (over choices of A and x) result in at least one other member of $S(x)$ being covered by R', unless the expected number of f-queries made during the computation of $C(s)$ for a random $s \in S(x)$ is $\Omega(\sqrt{n/p(n)})$. We will not use the fact that C happens to be a computational circuit; rather, we will treat it as simply a mapping that associates each input $s \in S(x)$ with a set of f-queries, and show the result combinatorially.

Theorem 3 *There exists an oracle A relative to which 1) there exist $2^{-O(p(n))}$-one-way permutations, but 2) any universal one-way hash function which compresses its input by a constant factor ε (that is, from n bits to $(1-\varepsilon)n$ bits) must invoke a one-way permutation an expected $\Omega(\sqrt{n/p(n)})$ times.*

Proof The structure of the proof is as follows: we first describe the separating oracle A formally and in detail, then show that it does not significantly help any algorithm attempting to invert the random permutation f on a random input. We then show how to convert the question of the existence of efficient UOWHFs relative to A into a purely combinatorial question about the existence of certain types of arrangements of colored balls in bins. Finally we prove a lemma in this combinatorial setting which, by the previous reduction, implies a lower bound on the efficiency of UOWHFs relative to A. The proof of this lemma is based in turn on the well-known "sunflower lemma" of Erdös and Rado ([ER60]).

Oracle description. The oracle A will "contain" a permutation f on strings of length n, and accept queries of the form (x, C), where C is a circuit description. The circuit described may contain special "f-gates" which denote a request to the oracle ("f-query") to compute f on the gate's input, as well as oracle gates ("A-gates") which denote submission of the gate's input as a normal query of A ("A-query"). Given such a circuit description, the oracle first verifies that the output length is at most a multiple $1-\varepsilon$ (for some fixed constant ε) of the input length. If so, it first selects a random *chosen set* $R \subset \{0,1\}^n$ by including each string independently with probability $2^{-p(n)}$, and outputs both $C(x)$ (the output of C on input x) and a value x' chosen uniformly from the set of possible inputs to C (not including x itself) for which the following two conditions hold: (1) C produces the same output on both x and x'; and (2) when computing $C(x')$, all inputs to f-gates are either members of R or else also inputs to an f-gate during the computation of C on input x. If this set is empty, then A outputs only $C(x)$. Finally, A appends to its output the *f-output set* $Q(C,x)$ of all the input-output pairs for all the f-queries made during the computation of $C(x)$. (A also outputs the f-output set $Q(C,x')$, if it exists). We define A to select each chosen set R permanently for a particular pair (x,C), so that repeated A-queries with the same input always produce the same output; A therefore computes a well-defined function.

We will consider the permutation f and the chosen sets R to be chosen randomly. More precisely, we define for every n a family $\{A_n\}$ of oracles of this type "containing" a permutation $f : \{0,1\}^n \rightarrow \{0,1\}^n$, with each using a table of the necessary length to determine its choices for every possible query circuit of size up to $n^{\omega(1)}$, and prove that for any polynomial-size circuit C, an A chosen uniformly at random from this family will find a collision with a random x with constant probability (over choices of A and x). It follows that this statement also holds for any distribution on polynomial-size (in n) circuits generated by selecting an $H_{h,n}$ uniformly at random from a UOWHF family H.

In fact, let A_f be the set of choices defining A apart from those made in defining the permutation f. We can therefore consider a random A as being constructed by choosing first an A_f and then an f, both uniformly at random. However, A_f may actually be chosen optimally for each C and x; hence this theorem implies the existence of an A_f such that the resulting A finds a collision for x with constant probability (over the choice of f).

Non-invertibility of f. We will first show that f remains a one-way permutation relative to A; in fact, we will show that polynomially many queries to A reveal a negligible amount of information, even to a computationally unbounded adversary, about a y chosen uniformly at random, given its image $f(y)$ under the randomly chosen permutation f. The intuitive reason for this is clear: A reveals no distinguishing information about preimage-image pairs in f whose preimages are not in either a (relatively small) chosen set or an f-output set for some A-query. Since the union of all chosen sets with all the f-output sets (for polynomially many queries) still forms a negligible fraction of all possible inputs, the probability is overwhelming that a random preimage y will fall outside this union; in that case, its image could equally well be any of the images not already paired with a preimage in the union.

Lemma 4 *Let the following be chosen uniformly at random: a permutation f; a set A_f of choices (apart from the permutation f) for an instance of the oracle A described above; and an input y into f. Then given the image $f(y)$, and the results of polynomially many A-queries chosen adaptively by an arbitrary computationally unlimited adversary, the optimal guess for y is correct only with probability $2^{-O(p(n))}$ (assuming $p(n) \in \omega(\log n)$).*

Proof Consider an oracle B which accepts inputs in the same form as A's (that is, in the form (x, C)), but simply returns the entire chosen set R for that query (as determined by A_f), together with R's members' images under f, as well as $C(x)$ and its associated f-output set. (That is, rather than select an x', B simply supplies all the necessary information for a computationally unlimited adversary to select its own x'.) Consider also an arbitrary computationally unlimited adversary given the image $f(y)$ of the preimage y chosen uniformly at random, and choosing polynomially many A-queries adaptively in order to guess y with optimal probability of correctness. Note that although the adversary's B-queries can themselves contain nested B-queries, it is always possible to order all of the queries so that earlier ones are not dependent on the results of later ones.

Now suppose that none of the B-queries before a particular B-query B_i has resulted in an f-query with input y. Let S_i be the union of the chosen sets for all B-queries before B_i, together with the set of all inputs into f-queries made prior to B_i. Let F be the set of all permutations on $\{0,1\}^n$ which are identical to f on S_i. Since S_i is still only a fraction $2^{-O(p(n))}$ of the set of possible inputs into f, the set of possible values of y is still $2^n(1 - 2^{-O(p(n))})$. Moreover, all those possible values are equally likely, since the results of the B-queries are the same regardless of which element of F (which contains equally many functions for each possible value of y) is the correct one. Hence the probability that S_{i+1} contains y is $2^{-O(p(n))}$. Extending the same reasoning, the probability that y is ever contained in a chosen set or the input to an f-query after polynomially many B-queries is $2^{-O(p(n))}$. It follows that the same holds for A.

Conversion to a combinatorial setting. We will now show that relative to a randomly chosen A, a collision can be found for a given input in any UOWHF construction if it averages fewer than $\Omega(\sqrt{n/p(n)})$ f-queries per input and compresses the input by some constant factor ε. All that is required in that case is to show that for any circuit C, there is a constant probability (over the choices of A and random input x) that at least one value in the colliding set of x under C generates f-queries in C that are all in either the chosen set or the f-output set for the A-query (x, C). If such an input value exists, then A will output it on input (x, C), and C will therefore not be useful in constructing a UOWHF relative to A.

The proof is purely combinatorial, treating C as an arbitrary function with arbitrary f-queries. If C compresses the input by εn bits, then a randomly chosen input x will with non-negligible probability collide with at least $2^{\varepsilon n}$ other inputs under C. Consider these $2^{\varepsilon n}$ inputs as colors, each of which is assigned to at most k balls; these are in turn placed arbitrarily (by an adversary, say) in one of 2^n buckets (representing the 2^n possible values which the inputs can take on). We will show that with high probability an f-output set of all the buckets containing at least one ball of a particular randomly chosen color, together with a chosen set of randomly chosen buckets selected independently with probability $2^{-p(n)}$, will contain all the balls of at least one other color, for $k \in O(\sqrt{n/p(n)})$ and $p \in \omega(\log n)$.

It follows that given a circuit C which averages $O(\sqrt{n/p(n)})$ f-queries per input, a randomly chosen x that collides with at least $2^{\varepsilon n}$ other inputs under C, and a random chosen set of inputs selected independently with probability $2^{-p(n)}$, A will with non-negligible probability output an x' that collides with x. (Note that if C averages fewer than z f-queries per input, and compresses by εn bits, then a randomly chosen input into C will with probability at least $2/3$ cause at most

$3z$ f-queries, and with probability at least $1 - 2^{\varepsilon n/2}$ collide with at least $2^{\varepsilon n/2}$ other inputs. Hence the only effect of variable numbers of f-queries and variable-size colliding sets is to alter some constants.)

Combinatorial lemma. The following lemma proves the required result:

Lemma 5 *For a set of balls of $2^{\varepsilon n}$ different colors, with k or fewer balls per color, placed arbitrarily into 2^n buckets, let R be a set of buckets chosen by including each bucket with probability $2^{-p(n)}$, and let Q be the set of all buckets containing a ball of color c (where c is chosen uniformly at random). Then there exists a constant $\delta > 0$ such that with constant probability (over the choices of R and c) $Q \cup R$ contains all the balls of some color other than c, as long as $k \leq \delta\sqrt{n/p(n)}$.*

Proof The proof uses the "sunflower" theorem of Erdös and Rado:

> **Lemma 6** *("sunflower lemma"; [ER60]) Let $\Delta = \{\Delta_1, ..., \Delta_\ell\}$ be a collection of sets such that for all $i \neq j$ and $i \neq j'$, $\Delta_i \cap \Delta_j = \Delta_i \cap \Delta_{j'}$ (we call such a collection a* sunflower *of size ℓ). Let $f(k, \ell)$ be the minimum cardinality for a collection of sets of size at most k such that it is guaranteed to contain as a subcollection a sunflower of size ℓ. Then $f(k, \ell) \leq (\ell - 1)^k k!$.*
>
> Now, set $\ell = \gamma p^{-k} + 1$ (for some constant γ), and let "bucket set" Δ_i be the set of buckets containing a ball of color i. Then each arbitrary collection of $\sigma = \gamma^k p^{-k^2} k!$ bucket sets contains a sunflower of size ℓ, by the above lemma. We can thus form sunflowers out of disjoint collections of bucket sets until fewer than σ bucket sets remain. The probability that c's bucket set (call it D) is not within one of the sunflowers is at most $\sigma/2^{\varepsilon n} \in 2^{-O(n)}$ (assuming a judicious choice of δ in Lemma 5). And if D is in one of the sunflowers, then the common intersection of all the bucket sets in D's sunflower is guaranteed to be in $Q \cup R$, and the sets are moreover disjoint apart from the common intersection. Hence the probability that a given bucket set from D's sunflower is contained in $Q \cup R$ is independent of whether any of the others is as well. That probability is bounded below by p^k; hence, the probability

that no bucket set is covered is at most $(1-p^k)^\ell \leq \alpha$ for some constant $\alpha < 1$ (assuming a judicious choice of γ).

4 The Oracle Separation and "Black Box" Constructions

The oracle A presented above can be used to show that any construction of a UOWHF which assumes only a generic $p(n)$-one-way permutation, treating it as a "black box" (i.e., an oracle) for the purposes of the construction, and compresses by a factor ε, must necessarily average at least $\Omega(\sqrt{n/p(n)})$ invocations of the one-way permutation during its computation. Consider, for instance, an oracle F which, for a given size input of which the first half of the input bits are ones, outputs the result of A on the latter half of the input, and otherwise, computes the one-way permutation f described above (which remains a one-way permutation even in the presence of A). A simple permutation-preserving trick (mapping inputs of the form $(11...1x, x, ..., x)$, for suitably many repetitions of x, to $(11...1x, A(x))$, and vice versa, for every x) can be used to turn F into a permutation oracle Π; Π preserves F's "one-wayness" (as long as most inputs still result in a simple computation of f) as well as F's feature of offering callers complete access to A (using polynomially larger-sized inputs). It follows that any proof of a more efficient UOWHF from a one-way permutation must implicitly assume that the permutation oracle is not Π (which can be used to find collisions in any hash function with insufficiently many calls to the one-way permutation). Hence the proof cannot apply to an absolutely arbitrary one-way permutation.

Note that we are modeling the one-way permutation primitive here as a single oracle answering arbitrary-length queries. It is common for "black box" constructions based on abstract primitives to represent the primitive as a family of oracles with fixed input and output lengths, rather than as a single oracle; this is normally reasonable because such constructions are typically relativizing, meaning that the constructions are no less provable in the presence of longer-length oracles in the same family. A black-box construction with a non-relativizing proof that did not permit the presence of longer-length oracles could, in principle, exist (although it is difficult even to imagine one); however, it would say nothing of practical significance, since any feasible instantiation of the one-way permutation would necessarily be implementable for any length which is polynomial in the original one. Hence the conclusions drawn here based on the model of the one-way permutation as a single oracle still apply to all practically relevant constructions.

5 Conclusions and Open Problems

The $\Omega(\sqrt{n/p(n)})$ bound obtained here may well not be optimal; a natural conjecture would be that any bound of the form $\Omega(n/\omega(p(n)))$ would hold. More careful analysis might yield a bound closer to the conjectured one. A more carefully constructed oracle might also allow for a provability result in the manner of [IR89], in which it is shown that any provable construction of a key exchange protocol based solely on a one-way permutation would automatically yield a proof that $P \neq NP$.

The result here differs from most previous oracle separations of cryptographic primitives in that it focuses on the efficiency, rather than the security, of potential constructions. (Another exception can be found in [Rud91], which separates relativized key exchange protocols by efficiency in terms of number of communication rounds.) There are several other primitives, such as digital signatures and pseudorandom generators, which are known to be provably constructible from one-way functions, but for which no truly efficient one-way-function-based constructions have been found. Perhaps relativized methods may shed light on the question of whether the known constructions of such primitives can be made efficient enough to be practical.

References

[BR97] M. Bellare and P. Rogaway, "Collision-Resistant Hashing: Towards Making UOWHFs Practical", Proc. CRYPTO '97, 1997.

[Dam87] I. Damgård, "Collision-Free Hash Functions and Public-Key Signature Schemes", Proc. EUROCRYPT '87, 1987.

[Dam89] I. Damgård, "A Design Principle for Hash Functions", Proc. CRYPTO '89, 1989.

[DP80] D. Davies and W. Price, "The Application of Digital Signatures Based on Public-Key Cryptosystems", Proc. 5th International Computer Communications Conference, 1980.

[ER60] P. Erdös and R. Rado, "Intersection theorems for systems of sets", J. London Math. Soc. **35** (1960), pp. 85–90.

[IR89] R. Impagliazzo and S. Rudich, "Limits on the Provable Consequences of One-Way Permutations", Proc. 21st Annual Symposium on Theory of Computing, 1989, pp. 44–61.

[Mer89] R. Merkle, "One Way Hash Functions and DES", Proc. CRYPTO '89, 1989.

[NIST94] National Institute of Standards and Technology, NIST FIPS PUB 186, "Digital Signature Standard", U.S. Department of Commerce, 1994.

[NY89] M. Naor and M. Yung, "Universal Hash Functions and their Cryptographic Applications", Proc. 21st Annual Symposium on Theory of Computing, 1989.

[Riv92] R. Rivest, "The MD5 Message Digest Algorithm", RFC 1321, 1992.

[Rom90] J. Rompel, "One-Way Functions Are Necessary and Sufficient for Digital Signatures", Proc. 22nd Annual Symposium on Theory of Computing, 1990.

[Rud91] S. Rudich, "The Use of Interaction in Public Cryptosystems", Proc. CRYPTO '91, 1991.

[Sim98] D. Simon, "Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?", Proc. EUROCRYPT '98, 1998.