Basing Weak Public-Key Cryptography on Strong One-Way Functions

Eli Biham[1], Yaron J. Goren[2], and Yuval Ishai[3]

[1] Technion, Israel. biham@cs.technion.ac.il.
[2] Rosetta Genomic, Israel. yaron.goren@gmail.com.[⋆]
[3] Technion, Israel and UCLA, USA. yuvali@cs.technion.ac.il.[⋆⋆]

Abstract. In one of the pioneering papers on public-key cryptography, Ralph Merkle suggested a heuristic protocol for exchanging a secret key over an insecure channel by using an idealized private-key encryption scheme. Merkle's protocol is presumed to remain secure as long as the gap between the running time of the adversary and that of the honest parties is at most *quadratic* (rather than super-polynomial). In this work, we initiate an effort to base similar forms of public-key cryptography on well-founded assumptions.
We suggest a variant of Merkle's protocol whose security can be based on the *one-wayness* of the underlying primitive. Specifically, using a one-way function of exponential strength, we obtain a key agreement protocol resisting adversaries whose running time is nearly quadratic in the running time of the honest parties. This protocol gives the adversary a small (but non-negligible) advantage in guessing the key. We show that the security of the protocol can be amplified by using a one-way function with a strong form of a hard-core predicate, whose existence follows from a conjectured "dream version" of Yao's XOR lemma. On the other hand, we show that this type of hard-core predicate cannot be based on (even exponentially strong) one-wayness by using a black-box construction.
In establishing the above results, we reveal interesting connections between the problem under consideration and problems from other domains. In particular, we suggest a paradigm for converting (unconditionally) secure protocols in Maurer's *bounded storage model* into (computationally) secure protocols in the random oracle model, translating storage advantage into computational advantage. Our main protocol can be viewed as an instance of this paradigm. Finally, we observe that a *quantum* adversary can completely break the security of our protocol (as well as Merkle's heuristic protocol) by using the quadratic speedup of Grover's quantum search algorithm. This raises a speculation that there might be a closer relation between (classical) public-key cryptography and quantum computing than is commonly believed.

1 Introduction

The fundamental cryptographic primitives and protocols can be roughly divided into two categories: "private cryptography" which includes private key encryption, pseudo-random generators, pseudo-random functions, bit commitment and digital signatures,

[⋆] Most of this work was done while at the Department of Computer Science, Technion.
[⋆⋆] Supported by ISF grant 1310/06, BSF grant 2004361, and NSF grants 0205594, 0430254, 0456717, 0627781, 0716835, 0716389.

and "public cryptography" which includes public key encryption, key agreement, oblivious transfer, and secure function evaluation. Since the existence of these primitives implies that P≠NP, given the current state of complexity theory we need to base it on unproven computational assumptions. These assumptions may turn out to be false; thus, basing primitives on the minimal possible assumptions has been put forward as one of the most important goals in cryptography.

The weakest assumption that is commonly used in cryptography is the existence of *one-way functions*; it is weakest in the sense that if such functions do not exist, none of the above primitives exist [15]. The existence of one-way functions implies the existence of all of the private cryptography primitives (cf. [16, 6] and references therein). In contrast, it is not known how to base public cryptography primitives on one-way functions. Obtaining such a construction is arguably one of the most intriguing open questions in cryptography. In addition to its fundamental importance, this question is also motivated by the big efficiency gap between the best current implementations of public primitives and the (much more efficient) implementations of private primitives. This gap is mostly due to the fact that current approaches for obtaining public primitives rely on *algebraic* intractability assumptions. Since the underlying algebraic objects are highly structured, there are sophisticated attacks that exploit this structure. Thus, the underlying objects must be very large in order to defeat known attacks. An additional, more recent, motivation for basing public cryptography on one-way functions is the advent of efficient *quantum* algorithms that break most (but not all) of the concrete algebraic intractability assumptions that currently underly public cryptography [22].

In light of the above, basing public-key cryptography on one-way functions can be viewed as a "holy grail" both from a theoretical and from a practical point of view. In fact, from the latter point of view even a heuristic construction based on a random oracle might be considered satisfactory (as the random oracle can often be replaced in practice by a sufficiently "structureless" function). However, a seminal result of Impagliazzo and Rudich [16] suggests that standard methods cannot be used to realize such constructions. Specifically, this result rules out the possibility of a *black-box* construction based on a one-way permutation (see also [20]). Furthermore, the result of [16] shows that a provable construction of a public-key primitive based on a random permutation oracle is unlikely to be found, as it would imply a proof that P≠NP.

Weak public-key cryptography. An implicit assumption in the last statements is that the gap between the resources of the honest parties and those of the adversary must be super-polynomial. It is natural to relax this assumption and consider a weaker variant of public-key cryptography, where the resource gap between the adversary and the honest parties is bounded by some *fixed* polynomial. Such a weaker form of public-key cryptography has a similar qualitative flavor as standard public-key cryptography, and might be relevant to practice. Indeed, even with a quadratic resource gap, the *ratio* between the amount of time required by the adversary and that required by honest parties grows linearly with the computing power. Thus, security gets better with technology. In this work, we study the possibility of basing such weak public-key cryptography on one-way functions and related primitives.

Merkle's puzzles. Our point of departure is the pioneering work of Merkle [19], who proposed the following protocol for secret key agreement over public channels. Merkle's protocol involves two honest parties, Alice and Bob. It relies on the ability to efficiently create "puzzles" which encapsulate a value chosen by the puzzle creator and require a "moderate" amount of time T to be solved by another party. The protocol proceeds by letting Alice pick a large number S of random pairs (k_i, id_i) and send to Bob S puzzles encapsulating these pairs. Bob picks a random puzzle r and, after spending time T solving it, obtains a pair (k_r, id_r). It then sends id_r to Alice. Now both parties have a common key k_r. The time spent by Alice in this protocol is roughly S (assuming that a puzzle can be generated at a unit cost), and the time spent by Bob is roughly T. However, from the point of view of an external eavesdropper Eve, r remains secret. Thus, the intuition is that Eve will need to solve $S/2$ puzzles on average, spending $\Omega(ST)$ time, before she can learn k_r. Setting $S = T$, both Alice and Bob have a quadratic advantage over Eve. Merkle suggested a heuristic implementation of the puzzles using a weakened version of a private-key encryption scheme, where solving the puzzle amounts to exhaustively searching over a (moderately sized) key space.

Trying to instantiate the puzzles in the above protocol using a standard (semantically secure) private-key encryption scheme is problematic for several reasons. First, an implicit assumption that underlies the security of the protocol is that there is a sharp bound T between the maximal time required by honest parties to *completely* solve a puzzle and the minimal time required by an adversary to gain *some* information about the solution. One might try to achieve this goal by requiring the encryption to have "exponential strength" in its key size. However, it is not clear how to realize such a strong primitive based on (even strong versions of) low-level primitives such as a one-way function. A second problem is that the security of the resulting protocol seems to rely on the assumption that the adversary has no better strategy for recovering the key k_r than by trying to solve the puzzles one by one until finding the one that contains s_r. Again, this is an unsubstantiated assumption in a complexity-based cryptography.

1.1 Our Contribution

The question whether *weak* public key cryptography can be based on one-way functions, or some variation of them, is largely unexplored. Our goal is to understand what kinds of weak public key cryptography are possible and under what assumptions.

We start by suggesting a variant of Merkle's protocol which admits a simple proof of security in the random oracle model. In this protocol, each party (independently) evaluates a random *permutation* on a random set of inputs whose size is roughly the square root of the domain size, and the parties communicate the set of the outputs of these evaluations. By the birthday paradox, the two sets of outputs intersect with high probability, and the preimage of this intersection can be used to extract a common key. (The above protocol can be viewed as based on a similar protocol of Cachin and Maurer [2] in the *bounded storage model* – see below.)

We then show that the random permutation oracle in this protocol can be instantiated with an exponentially strong one-way permutation (OWP), or even an exponentially strong 1-1 OWF, yielding a key agreement protocol with a polynomial gap between the bounded parties and the adversary. Specifically, if the OWP is secure against adversaries that run in time $2^{(1-\delta)n}$, the protocol is secure as long as the running time of the adversary is less than the running time of the honest parties to the power of $2 - 2\delta$. Thus, we approach quadratic security as δ tends to 0. Towards obtaining a similar result under any one-way function, we show a way for transforming an exponentially strong OWF into a *family* of exponentially strong OWFs that are "almost" 1-1. (We stress that this transformation inherently relies on the exponential strength of the underlying OWF; its analysis gives a general method for redistributing the hardness of OWFs which may be of independent interest.) Using this transformation we obtain a similar key agreement protocol based on an exponentially strong OWF.

On the existence of strong one-way functions. Our protocols rely on one-way functions whose strength goes beyond the birthday paradox bound of $2^{n/2}$. The existence of such OWFs can be regarded as a very mild assumption from a cryptanalytic point of view. For instance, an explicit attack against AES that runs in time $2^{0.9n}$, where n is the key size, would be considered as indicating a major vulnerability. Exponentially strong OWFs were recently exploited in several cryptographic contexts. In the context of program obfuscation, Wee [23] uses a OWF with a form of exponential strength which is even stronger than ours in terms of the ratio between the adversary's time bound and its success probability. OWFs with milder forms of exponential strength were recently employed for constructing pseudorandom generators [14, 11, 4]. It should be noted that given generic time-space tradeoffs for inverting functions [12, 5], one cannot expect a *fixed* function (rather than a collection of functions) to be *non-uniformly* one-way with a very good exponential strength (say, better than $2^{2n/3}$). Thus, in the context of this work one should either restrict adversaries to be *uniform*, or alternatively rely on a *collection* of strong one-way functions.

On reducing the adversary's advantage. The key agreement protocols described above allow the adversary to gain an inverse polynomial advantage in guessing the secret key. This type of insecurity may be viewed as reasonable in the context of weak public-key cryptography, but it is still desirable to obtain the standard notion of security with negligible advantage with respect to a weaker class of adversaries. Unfortunately, known techniques for converting weak key agreement to strong one (e.g., those of Holenstein [13]) do not seem sufficient for this purpose. We show that the security of the protocol can be boosted to allow only a negligible advantage if one assumes the underlying primitive to have a strong form of a hard-core predicate [8] which we call a *multi-source hardcore predicate* (MSHCP). Roughly speaking, an MSHCP applies a predicate to several inputs, such that an (exponentially strong) adversary can only predict the value of the predicate on *independently chosen* inputs with an advantage that is negligible *in the size of the input domain*. (This should be contrasted with a standard hard-core predicate in which the predicate is applied to a single input, and it is only guaranteed that the advantage is negligible in the *bit-length of the input*.) We show that the existence of an MSHCP follows from a conjectured "dream version" of Yao's XOR lemma (a close variant of a conjecture appearing in [9]). On the other hand, we show that in contrast to standard hard-core predicates, the existence of an MSHCP cannot be based on (even exponentially strong) one-wayness by using a black-box construction.

Our results reveal some interesting and perhaps unexpected connections between the problem under consideration and problems from other domains.

Relation with the Bounded Storage Model. In Maurer's *bounded storage model* (BSM) [18], it is assumed that a large random source is transmitted, out of which the adversary can only store a limited amount of information. Viewing the random source as an oracle, the transmission of the random source can be replaced by local computation. In terms of security, the resulting model is incomparable to the original BSM: the adversary here is weaker in that it can only access "physical" bits of the source by querying the oracle (rather than store an arbitrary function of the source), but it is stronger in the sense that it is allowed access to the source even *after* the execution of the protocol. To get around the latter problem, a natural approach is to code the source in the *image* of the oracle. That is, the evaluation of the oracle f at point x gives a pair (i, b) indicating that the ith bit of the source is b. When the honest parties in the BSM protocol only need to access the source at *random* locations, the protocol can still be efficiently implemented using the random oracle. Our main protocol can be viewed as applying this conversion paradigm to the BSM protocol from [2]. A similar transformation can be applied to the oblivious transfer protocol from [3] to yield an oblivious transfer protocol (with quadratic security) in the random oracle model.

Relation with quantum computing. Finally, we observe that a *quantum* adversary can completely break the security of our protocol (as well as that of Merkle's heuristic protocol) by using the quadratic speedup of Grover's quantum search algorithm [10]. Thus, the two most prominent examples for speedup by quantum algorithms – the strong speedup of Shor's algorithm and the weaker speedup of Grover's algorithm – seem to be "tailored" to break the two main types of public-key cryptosystems – strong ones based on number-theoretic assumptions[4] and weak ones based on Merkle's technique. While this can be dismissed as a pure coincidence, it also raises the interesting speculation that there might be a closer relation between (classical) public-key cryptography and quantum computing than is commonly believed. This speculation may be supported by the relative scarcity of useful algorithms in the two domains.

It is important to stress that the quadratic speedup that can be achieved using Grover's algorithm is by no means universal, and applies only in scenarios that involve *parallel* search. An interesting problem left open by our work is that of obtaining a weak key agreement protocol, even in the random oracle model, that resists this kind of quantum attack. A natural approach for achieving this is by obtaining efficient implementations of puzzles that resist parallel search attacks. A similar problem was considered by Boneh and Naor [1] in the context of timed commitments. However, the only known implementations of this primitive rely on number-theoretic assumptions that do not resist a quantum attack. The possibility of implementing such "non-parallelizable" puzzles using a one-way function, or even a random function, remains open.

Organization. The remainder of this paper is organized as follows. Following some preliminaries, in Section 3 we describe a key agreement protocol based on a random function. The protocol resists adversaries whose running time is nearly quadratic in the running time of the honest parties. In Section 4 we replace the random function with an exponentially strong one-way permutation, and in Section 5 we show how to base a

[4] One should note in this context that we *do* have candidates for strong public-key cryptosystems that resist quantum attacks, mostly ones based on lattice problems and error-correcting codes. However, because of the strong algebraic structure of the underlying computational problems, the existence of efficient quantum algorithms for these problems does not seem unlikely.

variant of this protocol on an exponentially strong one-way function. Some details and proofs that were omitted from this version can be found in the full version.

2 Definitions

In contrast to conventional cryptography, in this work we assume the resource gap between the honest parties and the adversary to be bounded by a fixed polynomial. This requires us to introduce an "exact" variant for some common definitions and to set a concrete model of computation which is sensitive to such gaps. We use a RAM model (e.g., a "log-cost" RAM) as our default model of computation, for both honest parties and adversaries. A $T(n)$-bounded algorithm is an algorithm whose running time on input of length n is bounded by $T(n)$.

Our results are stated for uniform adversaries, but are valid for non-uniform adversaries as well. In this case the bound on the running time serves also as a bound on the size of the advice string given to the adversary. Specific differences between the results for uniform and non-uniform adversaries will be discussed when relevant.

Notation. We write $f(n) = \widetilde{O}(g(n))$ if there exists some constant c such that $f(n) = O(g(n)\log^c(g(n)))$. We say that a function $\epsilon(\cdot)$ is negligible and denote such a function by $neg(\cdot)$ if for any constant c, $\epsilon(n) < \frac{1}{n^c}$ for sufficiently large n. We say that $\epsilon(\cdot)$ is bounded away from c if $\epsilon(n) \le c - 1/p(n)$ for some polynomial p and all sufficiently large n. We denote by U_n the uniform distribution over $\{0,1\}^n$. By IP we denote the modular inner product function defined by $IP(x,r) = \sum_{i=1}^{n}(x_i \cdot r_i)$ mod 2.

2.1 Key Agreement Protocols

An $l(\cdot)$-bit key agreement protocol is an interactive protocol in which Alice and Bob receive a security parameter k, exchange messages over a public channel and each output a key in $\{0,1\}^{l(k)}$. Throughout the paper we deal only with 1-bit key agreement as defined below. Our protocols can be extended to $l(k)$-bit key agreement for any $l(k) \le \mathrm{polylog}(k)$ with similar asymptotic parameters by independent repetition. (A longer key will reduce the polynomial advantage of the honest parties.) Such an l-bit key can then be used to encrypt longer messages using a (conventional) symmetric encryption scheme. We note that the key agreement protocols presented in this paper are limited to two rounds. Hence what we achieve can be viewed as a (weak) *public key encryption* scheme, where the first message serves as the public key.

Definition 1 (Key agreement). *A protocol (Alice, Bob) is a* (d,ϵ)-secure key agreement protocol *if the following conditions hold:*

- Correctness: *Alice and Bob are* $\widetilde{O}(k)$*-bounded and they output the same bit except for a failure probability* $\delta(k) = neg(k)$.
- Security: *For any constant* $d' < d$ *and any* $O(k^{d'})$*-bounded adversary, for sufficiently large* k *the probability that the adversary guesses Bob's output on a random transcript of the protocol is bounded by* $\frac{1}{2} + \epsilon(k)$.

We say that the protocol has quadratic security *if it is* (d,ϵ)*-secure for* $d = 2$ *and some negligible* $\epsilon(\cdot)$.

2.2 Strong One-Way Functions and Hard-Core Predicates

Definition 2 (One-way function). *An efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$ is a (T, ϵ)* one-way function *if for any $T(n)$-bounded adversary A and for all sufficiently large n, $Pr_{x \in U_n}[f(A(1^n, f(x))) = f(x)] < \epsilon(n)$. If in addition f is a permutation we say that it is a (T, ϵ)* one-way permutation. *If f is (T, ϵ) one-way with $\epsilon(n) \leq \frac{1}{16}$, we say that it is a $T(n)$* one-way function.

We note that a standard one-way function is $(n^c, \frac{1}{n^c})$ one-way for every constant $c > 1$.

Definition 3 (Hard-core predicate). *An efficiently computable function $h : \{0,1\}^* \to \{0,1\}$ is a (T, ϵ)* (randomized) hard-core predicate *for f if for any $T(n)$-bounded adversary A, for sufficiently large n,*

$$Pr_{x \in U_n, r \in U_n}[A(f(x), r) = h(x, r)] < 1/2 + \epsilon(n).$$

The following definition generalizes the concept of hard-core predicates to allow the predicate to depend on several pre-images.

Definition 4 (Multi-source hard-core predicate). *A polynomial time computable function $H : \{0,1\}^* \to \{0,1\}$ is a (T, ϵ)* multi-source (randomized) hard-core predicate (MSHCP) *for f if there exist two polynomials $t(\cdot)$ and $s(\cdot)$ such that for any $T(n)$-bounded adversary A and all sufficiently large n,*

$$Pr_{x_1 \ldots x_{t(n)} \in U_n, r \in U_{s(n)}}[A(1^n, f(x_1) \ldots f(x_{t(n)}), r) = H(x_1 \ldots x_{t(n)}, r)] < 1/2 + \epsilon(n).$$

If H is a (T, ϵ) MSHCP with $\epsilon(n) = neg(2^n)$ we say that it is a strong MSHCP.

Note that a strong MSHCP can be guessed only with an advantage that is negligible in the size of the input domain. This is not possible with standard hard-core predicates since a single invocation of f is enough for finding a pre-image with probability which is the inverse of the input domain size. We also note that it is easy to show that a *random* function has a strong MSHCP; however, its security does not seem to follow from one-wayness alone. A relevant black-box separation is given in the full version.

3 Key Agreement in the Random Oracle Model

We describe a variant of Merkle's key agreement protocol in which all parties have access to a random function oracle, and show that adversaries whose running time is nearly quadratic in the running time of the honest parties can only have a negligible advantage in guessing the agreed key. For simplicity of presentation we assume that the function which the oracle computes is chosen uniformly from some set of functions *after* the adversary has been set, and in this scenario we bound the adversary's advantage in guessing the key. This result can be extended, using a standard argument [16], to show that the protocol is secure against any uniform adversary when the oracle is set to a specific function from the set, with probability 1 over the choice of functions. Alternatively, the protocol is secure against non-uniform adversaries when the function is chosen after the adversary is set (i.e. no single non-uniform adversary can break the protocol for a significant fraction of the functions).

We start with a protocol which uses an oracle to a random permutation and a random predicate and then extend the result for the case of a random function.

The ROM protocol: For a security parameter k we use an oracle to a random permutation $f : [k^2] \to [k^2]$ and a random predicate $h : [k^2] \to \{0, 1\}$. We set a minimal intersection size parameter $l(k)$ to be $l(k) = \frac{1}{2}\log^2(k)$.

- Alice chooses a random set $\mathcal{A} \subset [k^2]$ of size $k \cdot \log k$, queries the oracle on these inputs and sends $f(\mathcal{A}) = \{f(a)|a \in \mathcal{A}\}$ to Bob.
- Bob chooses a random set $\mathcal{B} \subset [k^2]$ of size $k \cdot \log k$ and queries the oracle on these inputs. If $|f(\mathcal{A}) \cap f(\mathcal{B})| < l(k)$ Bob aborts and both parties output random values. Otherwise, Bob randomly chooses $l(k)$ common outputs $c_1, \ldots, c_{l(k)} \in f(\mathcal{A}) \cap f(\mathcal{B})$ and sends them to Alice.
- Alice and Bob find the common inputs $s_1, \ldots, s_{l(k)} \in \mathcal{A} \cap \mathcal{B}$ such that $f(s_i) = c_i$ and output $\bigoplus_{i=1}^{l(k)} h(s_i)$.

In the full version we prove that the above protocol has quadratic security. We also show a similar protocol which uses a random function instead of a permutations and a predicate and prove the following theorem.

Theorem 1. *Given an oracle to a random function $f : \{0, 1\}^* \to \{0, 1\}$, there exists a key agreement protocol with quadratic security.*

4 Key Agreement from One-Way Permutations

In order to construct key agreement from one-way permutations we replace the random permutation and random predicate used in the ROM protocol with an exponentially strong one-way permutation and a hard-core predicate. The analysis is divided into two parts: first we show how to construct a key agreement protocol from a one-way permutation with an MSHCP and then we show how to construct an MSHCP for any one-way function. The maximal possible advantage in guessing the MSHCP determines the advantage the protocol allows adversaries in guessing the key. Using a conjectured dream XOR lemma we construct a strong MSHCP (in which the maximal advantage is negligible in the size of the input domain) and hence key agreement protocols in which the adversary's advantage is negligible. Without this conjecture we do not know how to construct strong MSHCPs, but can get a weaker MSHCP that suffices for limiting the adversary's advantage to $1/\text{poly}(k)$. In the full version we show that the limitation on the strength of the MSHCP is inherent to black-box constructions.

The results in this section hold in both uniform and non-uniform settings, depending on the setting in which the permutations are assumed to be one-way. However, it follows from generic time-space tradeoffs for inverting functions [12, 5] that one cannot expect a *fixed* function (rather than a collection of functions) to be non-uniformly one-way with a very good exponential strength. In order to achieve meaningful results in the non-uniform case, one may use a collection of one-way functions in the protocol described in Section 5.4. Finally, we note that the results of this section can be generalized to any 1-1 one-way function.

4.1 Key Agreement from a One-Way Permutation with an MSHCP

We use a variant of the ROM protocol in which the random permutation is replaced by a one-way permutation and the random predicate is replaced by an MSHCP.

The OWP protocol: For a security parameter k we set $n = 2 \cdot \log k$ (i.e. $k = 2^{n/2}$) and use a one-way permutation $f : \{0,1\}^n \to \{0,1\}^n$ for which H is a (T, ϵ) MSHCP with $T = 2^{n(1-\delta)}$. We set the minimal size of the intersection to be $l(k) = t(n)$ where $t(\cdot)$ is the number of inputs for H as in Definition 4.

– Alice chooses a random set $\mathcal{A} \subset \{0,1\}^n$ of size $k\sqrt{2l(k)}$, applies f to these inputs and sends $f(\mathcal{A}) = \{f(a) | a \in \mathcal{A}\}$ to Bob.
– Alice also sends Bob a random string $r \in \{0,1\}^{s(n)}$, where $s(\cdot)$ is the size of the random input for H as in Definition 4.
– Bob chooses a random set $\mathcal{B} \subset \{0,1\}^n$ of size $k\sqrt{2l(k)}$ and applies f to these inputs. If $|f(\mathcal{A}) \cap f(\mathcal{B})| < l(k)$ Bob aborts and both parties output random values. Otherwise, Bob randomly chooses $l(k)$ common outputs $c_1, \ldots, c_{l(k)} \in f(\mathcal{A}) \cap f(\mathcal{B})$ and sends them to Alice.
– Alice and Bob find the common inputs $s_1, \ldots, s_{l(k)} \in \mathcal{A} \cap \mathcal{B}$ such that $f(s_i) = c_i$ and output $H(s_1, \ldots, s_{l(k)}, r)$.

Theorem 2 (Key agreement from a one-way permutation with an MSHCP). *For any constant $\delta < \frac{1}{2}$, if there exists a one-way permutation with a (T, ϵ) MSHCP such that $T = 2^{n(1-\delta)}$ then there exists a (d, ϵ)-secure key agreement protocol with $d = 2 - 2\delta$.*

Proof. The proof of correctness is the same as in the ROM protocol. The proof of security is by contradiction. Suppose that for some constant $d' < 2 - 2\delta$, an $O(k^{d'})$-bounded adversary A guesses the agreed bit with probability at least $\frac{1}{2} + \epsilon$ when given a random transcript of the protocol. We show how to use A to guess $H(x_1, \ldots, x_{l(k)}, r)$, given $f(x_1), \ldots, f(x_{l(k)})$ and r on random $x_1, \ldots, x_{l(k)} \in \{0,1\}^n$ and $r \in \{0,1\}^{s(n)}$. We create a random transcript of the protocol using the following procedure:

– Randomly choose a set $\mathcal{A} \subset \{0,1\}^n$ of size $k\sqrt{2l(k)} - l(k)$, and apply f to these inputs. Randomly interleave $f(x_1), \ldots, f(x_{l(k)})$ within the set $f(\mathcal{A})$ and use the result as the first part of Alice's message to Bob.
– Use the random string r as the second part of Alice's message to Bob.
– Use $f(x_1), \ldots, f(x_{l(k)})$ as Bob's message to Alice.

It is easy to verify that the result is indeed distributed identically to a random transcript created by Alice and Bob. We then apply A to the transcript and output the same bit as A does. By our assumption A's output is equal to $H(x_1, \ldots, x_{l(k)}, r)$ with probability at least $\frac{1}{2} + \epsilon$. The transcript can be created in time $\widetilde{O}(k)$ and since $\delta < \frac{1}{2}$, for large enough n the total running time is bounded by $k^{2-2\delta} = 2^{n(1-\delta)}$, contradicting the hardness of H. $\square$

4.2 Construction of an MSHCP for any One-Way Function

We first construct a (single-source) randomized hard-core predicate defined by $h(x, r) = IP(x, r)$ for a random r and prove its security using an exact version of the Goldreich-Levin lemma. Then we use h to construct an MSHCP defined by $H((x_1, \ldots, x_t), (r_1, \ldots, r_t)) = \bigoplus_{i=1}^{t} h(x_i, r_i)$ and prove its security using an exact version of Yao's XOR lemma.

Lemma 1 (Goldreich-Levin). *If f is a (T, ϵ) one-way function then $h(x, r) = IP(x, r)$ (where IP denotes inner product modulo 2) is a (T', ϵ') randomized hard-core predicate for f with $T'(n) = T(n) \cdot \frac{\epsilon^4}{n^3}$ and $\epsilon'(n) = 4\epsilon$.*

The lemma follows from the alternative version of Proposition 2.5.3 in [7]. A consequence of this lemma is that every $T(n)$ one-way function has a (T', ϵ') (randomized) hard-core predicate with $T' = T(n)/\text{poly}(n)$ and $\epsilon' = 1/4$.

Definition 5 (Hard predicate). *We say that $P : \{0,1\}^* \to \{0,1\}$ is a (T, ϵ) hard predicate if for any $T(n)$-bounded adversary A and all sufficiently large n, $Pr_{x \in U_n}[A(x) = P(x)] < 1/2 + \epsilon(n)$.*

Lemma 2 (Yao's XOR lemma). *If P is a (T, ϵ) hard predicate and it is possible to efficiently sample from the distribution $(U_n, P(U_n))$, then for any $\mu(n)$ and $t = \text{poly}(n)$, $P^{(t)}(x_1, \ldots, x_t) = \bigoplus_{i=1}^t P(x_i)$ is a (T', ϵ') hard predicate for $T' = T \cdot \frac{\mu^2}{\text{poly}(n)} - \text{poly}(n)$ and $\epsilon' = (2\epsilon)^t + \mu$.*

The lemma can be derived by a careful analysis of Levin's proof for Yao's XOR lemma given in [17, 9]; see full version. Combining Lemma 1 with Lemma 2 allows us to construct an MSHCP but there is an inherent limitation to the strength of the MSHCP which we may construct in this way. A (T', ϵ') hard predicate constructed using Lemma 2 has the property that $\epsilon' > \mu$ and $T' < T \cdot \mu^2$. For any (T, ϵ) hard-core predicate for an *efficiently computable* one-way function, $T = \widetilde{O}(2^n)$ since it is possible to invert the one-way function by an exhaustive search. For our key agreement protocol we need a (T', ϵ') MSHCP in which $T' = 2^{(1-\delta)n}$ for some $\delta < \frac{1}{2}$. Any (T', ϵ') MSHCP constructed under the above restrictions will have $\epsilon' > \mu > \sqrt{T'/T} > 2^{-n/4}$. The following conjecture allows us to construct a (T', ϵ') hard predicate with $\epsilon' = neg(2^n)$ while T' remains close to T, and thus can be used to construct a strong MSHCP.

Conjecture 1 (Dream XOR lemma). If P is a (T, ϵ) hard predicate for some ϵ that is bounded away from $\frac{1}{2}$ and it is possible to efficiently sample from the distribution $(U_n, P(U_n))$, then there exists a constant $c < 1$, a negligible $\mu(\cdot)$ and some $\eta(\cdot)$ which is bounded away from 1 such that for any $t = \text{poly}(n)$, $P^{(t)}(x_1, \ldots, x_t) = \bigoplus_{i=1}^t P(x_i)$ is a (T', ϵ') hard predicate for $T' = T \cdot 2^{-o(n)}$ and $\epsilon' = 2^{cn} \cdot \eta^t + \mu(2^n)$.

A similar "dream version" of Yao's XOR lemma was conjectured in [9] and it was observed that it can not be proved using a black-box analysis. (The variant appearing in [9] requires ϵ to be smaller but allows T to be smaller as well.) Theorem 3 states the parameters of MSHCP that can be obtained with and without the dream XOR lemma conjecture, while in the full version we show that a strong MSHCP cannot be obtained from a OWF using a black-box construction.
We apply Lemma 2 and Conjecture 1 with $t = \text{poly}(n)$ to the predicate defined in Lemma 1 to get the following result (smaller values of t suffice for the first two cases, but this does not improve the asymptotic result).

Theorem 3. *For any $\delta < 1$, every $2^{n(1-\delta)}$ one-way function has a (T, ϵ) MSHCP with the following T and ϵ:*

- $T = 2^{n(1-\delta)}/\text{poly}(n)$	$\epsilon = \frac{1}{\text{poly}(n)}$	*using* $\mu = O(\epsilon)$
- $T = 2^{n(1-\delta-\tau)}/\text{poly}(n)$	$\epsilon = 2^{-\tau n/2}$	*using* $\mu = O(2^{-\tau n/2})$
- $T = 2^{n(1-\delta)}/2^{o(n)}$	$\epsilon = neg(2^n)$	*assuming the dream XOR lemma*

It is easy to verify that Theorem 2 holds also when $T = 2^{n(1-\delta)}$ is replaced with $2^{n(1-\delta)}/2^{o(n)}$. Combining this with Theorem 3 we get the main result regarding the construction of key agreement from one-way permutations. It relates the strength of the underlying one-way permutation to the security of the key agreement protocol that can be constructed from it.

Corollary 1. *For any constant $\delta < \frac{1}{2}$ if there exists a $2^{n(1-\delta)}$ one-way permutation then there exists a (d, ϵ) secure key agreement protocol for the following d and ϵ:*

- $d = 2 - 2\delta$	$\epsilon = 1/\log^c k$	*for any constant* c
- $d = 2 - 2\delta - 2\tau$	$\epsilon = k^{-\tau}$	*for any* $\tau < 1 - \delta$
- $d = 2 - 2\delta$	$\epsilon = neg(k)$	*assuming the dream XOR lemma*

5 Key Agreement from One-Way Functions

We extend the result from the previous section to obtain weak key agreement from exponentially strong one-way functions. The main technical result in this section is a construction of a collection of one-way functions which is almost 1-1 from an exponentially strong one-way function which is not necessarily 1-1. The construction applies a restriction to the domain of the one-way function such that the restricted function usually remains one-way and is almost always 1-1. The resulting collection of one-way functions is then used to construct a key agreement protocol. The results in this section hold in both uniform and non-uniform settings, depending on the setting in which the functions are assumed to be one-way. We refer to the uniform setting by default.

5.1 Definitions

A collection of one-way functions is defined by a pair of functions G and F such that $G(1^n)$ generates a key i of length $l(n)$ which defines a function $f_i : \{0,1\}^n \to \{0,1\}^*$ and $F(i, x)$ computes $f_i(x)$.

Definition 6 (Collection of one-way functions). *Let $\mathcal{F} = \bigcup F_n$ be a collection of functions where $F_n = \{f_i : \{0,1\}^n \to \{0,1\}^* | i \in I_n\}$. We say that $\mathcal{F}$ is (T, ϵ)* one-way *if there exist two PPT algorithms G and F such that the following holds:*

1. Easy to compute: *There exists a polynomially bounded function $l(\cdot)$ such that the output of G on input 1^n is in the set $I_n \subseteq \{0,1\}^{l(n)}$. On input $i \in I_n$ and $x \in \{0,1\}^n$, $F(i, x) = f_i(x)$.*
2. Hard to invert: *For every $T(n)$-bounded adversary A, for all sufficiently large n's,*

$$Pr_{i \in G(1^n), x \in U_n}[f_i(A(1^n, i, f_i(x))) = f_i(x)] < \epsilon(n).$$

We say that $\mathcal{F}$ is $T(n)$ one-way *if it is (T, ϵ) one-way with $\epsilon \le 1/32$. We say that $\mathcal{F}$ is* almost 1-1 *if the probability that $f_i \in F_n$ is not 1-1 is bounded by 2^{-n}.*

We will be using a family of injective *length-increasing*, pairwise independent hash functions in order to restrict the domain of a one-way function and increase the probability that it is 1-1. The function $m(\cdot)$ in the definition below determines the length of the output of the hash functions relative to the input length.

Definition 7 **($m(\cdot)$ pairwise independent family of hash functions).** *Let* $\mathcal{H} = \bigcup H_n$ *be a collection of functions where* $H_n = \{h_i : \{0,1\}^n \to \{0,1\}^{m(n)} | i \in I_n\}$*. We say that* $\mathcal{H}$ *is a* $m(\cdot)$ pairwise independent family of hash functions *if there exist two PPT algorithms G and H such that the following holds:*

1. Easy to compute: *There exists a polynomially bounded function* $l(\cdot)$ *such that the output of G on input* 1^n *is in the set* $I_n \subseteq \{0,1\}^{l(n)}$*. On input* $i \in I_n$ *and* $x \in \{0,1\}^n$*,* $H(i, x) = h_i(x)$*.*
2. Pairwise independent: *For every* $x_1 \neq x_2 \in \{0,1\}^n$ *and* $y_1 \neq y_2 \in \{0,1\}^m$*,* $Pr_{h \in H_n}[h(x_1) = y_1] = \frac{1}{2^m}$ *and* $Pr_{h \in H_n}[h(x_2) = y_2 \mid h(x_1) = y_1] = \frac{1}{2^m - 1}$*.*

Definition 7 can be instantiated with the collection of functions of the form $h_{a,b}(x) = ax + b$ where $a, b \in$ GF(2^m), $a \neq 0$, both addition and multiplication are over GF(2^m), and every x is interpreted as a distinct element of the subfield GF(2^n). Definition 7 implies that each function from the collection must injective. Moreover, it also implies the following *balance* property that will be useful in our analysis: For every n and every $y \in \{0,1\}^{m(n)}$, $Pr_{h \in H_n}[\exists x$ such that $y = h(x)] = \frac{2^n}{2^m}$.

5.2 Restricted Exponentially Strong One-Way Functions are 1-1

We show that strong one-way functions do not have many collisions to begin with (Lemma 3), and that by restricting the domain of such a function we get a function which is 1-1 with high probability (Theorem 4).

Definition 8 **(Collision group of y relative to f).** $[y]_f = \{y' | f(y') = f(y)\}$

We denote the input length of the one-way function in the following lemma by m, as we will use n for the input length of the family of one-way functions that we construct.

Lemma 3 **(Exponentially strong one-way functions have few collisions).** *If* f *is a* (T, ϵ) *one-way function then there exists a polynomial* $t(\cdot)$ *such that for sufficiently large* m *and for every* $y \in \{0,1\}^m$*,* $|[y]_f| \leq 2^m \cdot \max\{2\epsilon(m), t(m)/T(m)\}$*.*

Proof. We give the proof for the uniform setting, for the non-uniform case a stronger version of the lemma can be proved using the fact that a pre-image for y^* defined below can be given to the algorithm as advice. Let $t(\cdot)$ be the time required for evaluation of f. Assume for contradiction that the statement does not hold for y^*. We first show a $T(m)$-bounded algorithm that finds a pre-image for $f(y^*)$ with probability $\frac{1}{2}$. Randomly choose $y_1, \ldots, y_{T/t} \in \{0,1\}^m$ and apply f to them. By the assumption $|[y^*]_f| > 2^m \cdot t/T$, hence for each i, $y_i \in [y^*]_f$ with probability at least t/T. Therefore the probability that there is no i for which $x_i \in [y^*]_f$ is at most $(1 - t/T)^{T/t} < \frac{1}{2}$.

By the assumption $|[y^*]_f| > 2^m \cdot 2\epsilon$, therefore $y \in [y^*]_f$ with probability at least 2ϵ. If indeed $y \in [y^*]_f$, finding a pre-image for $f(y)$ is the same as finding a pre-image for $f(y^*)$ and the algorithm described above will find a pre-image with probability at least $\frac{1}{2}$. The conclusion is that there exists a $T(m)$-bounded adversary which finds a pre-image for $f(x)$ on a random x with probability ϵ, contradicting the fact that f is a (T, ϵ) one-way function. □

Notation. From here on we let $f_h(\cdot) \stackrel{\text{def}}{=} f(h(\cdot))$.

Theorem 4 (Restricted exponentially strong one-way functions are 1-1). *If $\mathcal{H} = \bigcup H_n$ is an $m(\cdot)$ pairwise independent family of functions and f is a (T, ϵ) one-way function with $T(m) \geq 2^{\mu \cdot m}$ and $\epsilon(m) \leq 2^{-\mu \cdot m}$ for some $\mu > 0$, then the probability over $h \in H_n$ that f_h is not 1-1 is bounded by* $\text{poly}(m) \cdot 2^{2n - \mu m}$.

Proof. Fix $x_1 \neq x_2 \in \{0,1\}^n$. We bound the probability that f_h maps both inputs to the same image for a random $h \in H_n$.
$Pr_{h \in H_n}[f_h(x_1) = f_h(x_2)] = Pr_{h \in H_n}[h(x_2) \in [h(x_1)]_f]$

$$= \sum_{y_1 \in \{0,1\}^m} Pr_{h \in H_n}[h(x_1) = y_1] \cdot Pr_{h \in H_n}[h(x_2) \in [y_1]_f \mid h(x_1) = y_1]$$

$$= \sum_{y_1 \in \{0,1\}^m} Pr_{h \in H_n}[h(x_1) = y_1] \cdot \sum_{y_2 \in [y_1]_f} Pr_{h \in H_n}[h(x_2) = y_2 \mid h(x_1) = y_1]$$

$$\leq 2^m \cdot 2^{-m} \cdot 2^m \cdot \max\{2\epsilon, \text{poly}(m)/T\} \cdot \frac{1}{2^m - 1} \quad (1)$$

$$\leq 2^{-\mu \cdot m} \cdot \text{poly}(m) \quad (2)$$

where (1) follows from the pairwise independence of $\mathcal{H}$ and from Lemma 3, and (2) follows from the hypothesis of the theorem about ϵ and T. As there are 2^{2n} pairs of inputs in $\{0,1\}^n$, by a union bound the probability that h_f maps *any* two inputs of length n to the same output is bounded by $\text{poly}(m) \cdot 2^{2n} \cdot 2^{-\mu \cdot m}$. □

5.3 Restricted Exponentially Strong One-Way Functions are One-Way

We show that if f is a strong one-way function and $\mathcal{H}$ is a pairwise independent family of hash functions, then $\mathcal{F} = \{f_h \mid h \in \mathcal{H}\}$ is a collection of strong one-way functions. The idea is that if we have an algorithm that finds a pre-image for $z = f_h(x)$ given z and h for a random h and x, we can use it to find a pre-image for $z = f(y)$ for a random y in the following way. We randomly select sufficiently many functions $h \in \mathcal{H}$ so that one of them will have y in its range, and apply the inversion algorithm to z and each such h. If the algorithm succeeds, we get some x such that $f_h(x) = f(h(x)) = z$ and $h(x)$ is a pre-image of z under f. This approach gives the following theorem.

Theorem 5. *Suppose f is (T, ϵ) one-way and $\mathcal{H}$ is a $m(\cdot)$ pairwise independent family of functions. Then $\mathcal{F} = \{f_h \mid h \in \mathcal{H}\}$ is a (T', ϵ') one-way collection of functions for any T', ϵ' such that for all sufficiently large n the following conditions hold with $m = m(n)$:*

1. $T'(n) \leq \frac{1}{2} \cdot T(m)$

2. $\epsilon'(n) \geq 2 \cdot \epsilon(m)$
3. $\frac{T'(n)}{\epsilon'(n)} \leq \frac{1}{4} \cdot \frac{2^n}{2^m} \cdot \frac{T(m)}{\epsilon(m)}$
4. $T'(n) > m(n)^c$ *for every constant* c

Proof. Throughout the proof we view T' and ϵ' as functions of n and T and ϵ as functions of m (where m is itself a function of n). We omit m and n in order to simplify notation. Assume for contradiction that f is a (T, ϵ) one-way function, but $\mathcal{F}$ is not a (T', ϵ') one-way collection for $T, T', \epsilon, \epsilon'$ as stated in the theorem. By Definition 6 there exists a T'-bounded adversary A' such that for infinitely many n's $Pr_{h \in H_n, x \in U_n}[f_h(A'(1^n, f_h(x), h)) = f_h(x)] \geq \epsilon'$. We construct a T-bounded algorithm A such that for infinitely many m's $Pr_{y \in U_m}[f(A(1^m, f(y))) = f(y)] \geq \epsilon$, in contradiction with the assumption that f is (T, ϵ) one-way.

The algorithm A, described below, finds a pre-image for $z = f(y)$. On input $(1^m, z)$, A repeats the following steps $t = \frac{T}{2T'}$ times:

1. Choose a random $h \in H_n$.
2. Compute $x = A'(1^n, z, h)$
3. If $f_h(x) = z$ stop and output $h(x)$

Assuming $z = f(y)$, we define the random variables $A_i(y)$ and $B_i(y)$ for $i = 1, \ldots, t$ as follows: $A_i(y) = 1$ if in the i'th iteration there exists a pre-image for y under h, and $B_i(y) = 1$ if in the i'th iteration $f_h(x) = f(y)$. (Probabilities in these variables are taken over the random coins of A.) The following two claims allow us to complete the proof of Theorem 5.

Claim 1 $Pr_{y \in U_m}[\exists i : A_i(y) = 1] \geq \frac{\epsilon}{\epsilon'}$

Claim 2 *For every* i, $Pr_{y \in U_m}[B_i(y) = 1 \mid A_i(y) = 1] \geq \epsilon'$

By the above claims and the definitions of A_i and B_i, we can lower bound the probability that A finds a pre-image for $f(y)$.

$$\begin{aligned} Pr_{y \in U_m}[f(A(1^m, f(y))) = f(y)] &= Pr_{y \in U_m}[\exists i : B_i(y) = 1] \\ &\geq Pr[\exists i : A_i(y) = 1] \cdot Pr[B_i(y) = 1 \mid A_i(y) = 1] \\ &\geq \frac{\epsilon}{\epsilon'} \cdot \epsilon' \\ &= \epsilon \end{aligned}$$

A's running time in every iteration is bounded by $T' + m^c$ for some constant c. As there are $t = \frac{T}{2T'}$ iterations and $T' > m^c$ the total running time is bounded by $\frac{T}{2T'} \cdot (T' + m^c) < T$, contradicting the assumption that f is (T, ϵ) one-way. □

Proof of Claim 1. We show that for any fixed $y \in \{0,1\}^m$, $Pr[\exists i : A_i(y) = 1] \geq \frac{\epsilon}{\epsilon'}$ and the claim follows. By the balance property we have $Pr[A_i(y) = 1] = Pr_{h \in H_n}[\exists x$ such that $y = h(x)] = \frac{2^n}{2^m}$ for every $i = 1, \ldots, t$. We view $A_1, \ldots, A_t$ as $t = \frac{T}{2T'}$ independent experiments, each with success probability $\delta = \frac{2^n}{2^m}$. The probability that none of the experiments succeed can be bounded by $(1-\delta)^t < \frac{1}{t\delta+1}$. Therefore $Pr[\exists i : A_i(y) = 1] \geq 1 - \frac{1}{t\delta+1} = \frac{t\delta}{t\delta+1}$. If $t\delta > 1$, $Pr[\exists i : A_i(y) = 1] \geq \frac{1}{2} \geq \frac{\epsilon}{\epsilon'}$, otherwise, $Pr[\exists i : A_i(y) = 1] \geq \frac{t \cdot \delta}{2} = \frac{1}{4} \cdot \frac{2^n}{2^m} \cdot \frac{T}{T'} \geq \frac{\epsilon}{\epsilon'}$. In both cases the last inequality follows from the assumptions in the hypothesis of the theorem. □

Proof of Claim 2. We use the following notation:
$p_1 \triangleq Pr_{y \in U_m}[B_i(y) = 1 \mid A_i(y) = 1]$
$p_2 \triangleq Pr_{h \in H_n, y \in U_m}[f_h(A'(1^n, f(y), h)) = f(y) \mid \exists x : y = h(x)]$
$p_3 \triangleq Pr_{h \in H_n, x \in U_n}[f_h(A'(1^n, f_h(x), h)) = f_h(x)]$
By the definition of A_i and B_i we have $p_1 = p_2$ for every i, and by our assumption that $\mathcal{F}$ is not (T', ϵ') one-way, we have $p_3 \geq \epsilon'$. Since Claim 2 is that $p_1 \geq \epsilon'$ for every i, it remains to show that $p_2 = p_3$. We show this by proving that the pair (y, h) under the conditions in p_2 is distributed identically to the pair $(h(x), h)$ under the conditions in p_3. Fix some (y, h) such that y is in the range of h. We calculate the probability of getting this pair under both distributions. Under the conditions in p_2, y is chosen uniformly from a set of size 2^m and h is chosen uniformly from a set of size $\frac{2^n}{2^m}|H_n|$, altogether the probability is $\frac{1}{|H_n|2^n}$. Under the conditions in p_3, h is chosen uniformly from a set of size $|H_n|$ and x is chosen from a set of size 2^n. Since h is 1-1, the probability for getting $y = h(x)$ is 2^{-n}, thus the overall probability is again $\frac{1}{|H_n|2^n}$. □

The above construction allows us to 'redistribute' the hardness of a one-way function. For example a (T, ϵ) one-way function which is strongly secure (small ϵ) against weak adversaries (small T) can be used to construct a (T', ϵ') one-way function family $\mathcal{F}$ that is weakly secure against strong adversaries. The conditions in the theorem give us the boundaries of the possible redistribution. Condition 1 limits the maximal gap between T' and T. Condition 2 does the same for ϵ. Both conditions are easy to satisfy when $m(\cdot)$ is large. Condition 3 defines the loss in the time over success ratio caused by the transformation. The loss is bigger when $m(\cdot)$ is large. For $m = c \cdot n$, if $\frac{T(n)}{\epsilon(n)} = 2^{n(1-\delta)}$ we will get $\frac{T'(n)}{\epsilon'(n)} < 2^{n(1-c\delta)}$.

Corollary 2. *If there exists a (T, ϵ) one-way function with $T, \epsilon^{-1} \geq 2^{m/3}$ and $T/\epsilon \geq 2^{m(1-\delta)}$ then there exists a collection of $2^{n(1-10\delta)}$ one-way functions which is almost 1-1.*

Proof. By applying Theorem 4 to the family $\mathcal{H}$ of $m(\cdot)$ pairwise independent family of hash functions with $m = 9n$ and a one-way function f, we get a family $\mathcal{F} = \{f_h | h \in \mathcal{H}\}$ which is 1-1, except for probability at most 2^{-n}. $\mathcal{F}$ is $2^{n(1-10\delta)}$ one-way if it is (T', ϵ') one-way with $T' = 2^{n(1-10\delta)}$ and $\epsilon' = 1/32$ and it remains to verify that for sufficiently large n, the conditions of Theorem 5 are fulfilled.

1. $T'(n) = 2^{n(1-10\delta)} < \frac{1}{2} \cdot 2^{3n} = \frac{1}{2} \cdot 2^{m/3} \leq \frac{1}{2} \cdot T(m)$
2. $\epsilon'(n) = \frac{1}{32} > 2 \cdot 2^{-m/3} \geq 2 \cdot \epsilon(m)$
3. $\frac{T'(n)}{\epsilon'(n)} = 32 \cdot 2^{n(1-10\delta)} < \frac{1}{4} \cdot 2^{n(1-9\delta)} = \frac{1}{4} \cdot 2^{n-\delta m} = \frac{1}{4} \cdot \frac{2^n}{2^m} \cdot 2^{m(1-\delta)} \leq \frac{1}{4} \cdot \frac{2^n}{2^m} \cdot \frac{T(m)}{\epsilon(m)}$
4. $T'(n) = 2^{n(1-10\delta)} > (9n)^c = m^c$ □

We note that in the non-uniform setting the parameters in the corollary can be improved by using a stronger version of Lemma 3.

5.4 Key Agreement from a Collection of 1-1 One-Way Functions

We show a key agreement protocol which is based on a collection of one-way functions $\mathcal{F}$ which is almost 1-1. The protocol is similar to the one described for constructing key agreement from a one-way permutation. However, there are two obstacles which prevent us from directly applying the previous protocol to a random $f_i \in \mathcal{F}$. First, unlike one-way permutations, $f_i \in \mathcal{F}$ is not always 1-1 and hence Alice and Bob may have different outputs. Second, since $\mathcal{F}$ is hard to invert only on average, it is possible that a specific function $f_i \in \mathcal{F}$ is easy to invert. We use standard techniques for amplifying both correctness and security. Specifically, we begin by describing a basic protocol which in itself lacks in both security and in correctness. By combining several copies of the basic protocol we create an intermediate protocol which is secure but has a big error probability. By combining several copies of the intermediate protocol we get the final protocol which is both secure and correct. We note that the copies of the basic protocol can be run concurrently and hence the final protocol remains a two message protocol.

The basic protocol. Let $\mathcal{F} = \bigcup F_n$, be a collection of $2^{n(1-\delta)}$ one-way functions which is almost 1-1. For a security parameter 1^k, we set $n = 2 \cdot \log k$. In each copy of the protocol, Alice chooses a random $r \in \{0,1\}^n$, a random index $i \in I_n$ which defines a function $f_i \in F_n$ and a random set $\mathcal{A} \subset \{0,1\}^n$ of size $k \cdot logk$. She applies f_i to the inputs in $\mathcal{A}$, and sends the outputs i and r to Bob. Bob randomly chooses a similar set of inputs $\mathcal{B}$, and applies f_i to them. If $f(\mathcal{A}) \cap f(\mathcal{B}) = \emptyset$, he aborts and both parties output random values; otherwise, he randomly chooses a common output $c \in f(\mathcal{A}) \cap f(\mathcal{B})$ and sends c to Alice. Alice and Bob each identify a source for the common input, $x_A \in \mathcal{A}$ and $x_B \in \mathcal{B}$, so that $f_i(x_A) = f_i(x_B) = c$. Their outputs are $s_A = IP(x_A, r)$ and $s_B = IP(x_B, r)$ respectively.

The intermediate protocol. We denote Alice and Bob's outputs in the i'th copy of the basic protocol by s_A^i and s_B^i. The intermediate protocol consists of $l = \mathrm{polylog}(k)$ copies of the basic protocol, where Alice and Bob's outputs are $S_A = \bigoplus_{i=1}^{l} s_A^i$ and $S_B = \bigoplus_{i=1}^{l} s_B^i$ respectively.

The final protocol. We denote Alice and Bob's outputs in the i'th copy of the intermediate protocol by S_A^i and S_B^i. The final protocol consists of $l = \mathrm{polylog}(k)$ copies of the intermediate protocol with the following addition to Bob's messages. Bob chooses a random $S \in \{0,1\}$ and for each copy of the intermediate protocol sends $S \bigoplus S_B^i$ to Alice. Bob's output is S and Alice's output is $MAJ\{S \bigoplus S_A^i \bigoplus S_B^i\}_{i=1}^{l}$.

A straightforward analysis of the final protocol (appearing in the full version) gives the following theorem:

Theorem 6 (Key agreement from a collection of one-way functions which is almost 1-1). *For any constant $\delta < 1/2$, if there exists a collection of $2^{n(1-\delta)}$ one-way functions which is almost 1-1, then there exists a (d, ϵ) secure key agreement protocol for the following d and ϵ:*

- $d = 2 - 2\delta$	$\epsilon = 1/\log^c k$	*for any constant c*
- $d = 2 - 2\delta - 2\tau$	$\epsilon = k^{-\tau}$	*for any $\tau < 1 - \delta$*
- $d = 2 - 2\delta$	$\epsilon = neg(k)$	*assuming the dream XOR lemma*

Combining Theorem 6 with Corollary 2, we get our main result on weak public-key cryptography from strong one-way functions.

Corollary 3 (Key agreement from one-way functions). *For any constant $\delta < 1/10$, if there exists a (T, ϵ) one-way function with $T, \epsilon^{-1} \geq 2^{m/3}$ and $T/\epsilon \geq 2^{m(1-\delta)}$ then there exists a (d, ϵ) secure key agreement protocol for the following d and ϵ:*

- $d = 2 - 20\delta$	$\epsilon = 1/\log^c k$	*for any constant c*
- $d = 2 - 20\delta - 2\tau$	$\epsilon = k^{-\tau}$	*for any $\tau < 1 - 10\delta$*
- $d = 2 - 20\delta$	$\epsilon = neg(k)$	*assuming the dream XOR lemma*

6 Conclusions and Open Problems

We established the feasibility of basing weak public-key cryptography on strong, but arguably reasonable, forms of one-way functions. We leave open the possibility of basing weak public-key cryptography on standard (polynomially strong) one-way functions, as well as the possibility of amplifying the security of our protocols without relying on a conjectured dream version of Yao's XOR Lemma. Finally, an interesting open question that was already discussed in the Introduction is the possibility of resisting quantum attacks in our setting. The discussion in Section 1.1 referred to the case where the honest parties are classical and the adversary is quantum. If the honest parties are quantum (and can therefore also exploit the quadratic speedup of Grover's algorithm), it seems possible to retain some of the efficiency gap between the honest parties and the adversary. Setting $T = S^2$ in the description of Merkle's protocol from Section 1, honest quantum parties can run in time $O(T)$ whereas a quantum adversary needs to run in time $\Omega(T^{3/2})$. The optimality of this gap, as well as the possibility of basing it on (quantum) one-way functions, remain to be further studied.

Acknowledgement. We thank the anonymous referees for helpful suggestions and comments.

References

1. D. Boneh and M. Naor. Timed Commitments. CRYPTO 2000: 236-254.
2. C. Cachin and U. Maurer. Unconditional Security Against Memory-Bounded Adversaries. In Advances in Cryptology - CRYPTO '97, pages 292-306, 1997.
3. Y. Ding, D. Harnik, R. Shaltiel and A. Rosen. Constant-Round Oblivious Transfer in the Bounded Storage Model. Theory of Cryptography — TCC '04, volume 2951, Cambridge, MA, USA, February 2004.
4. B. Dubrov and Y. Ishai. On the randomness complexity of efficient sampling. Proceedings of STOC 2006, pages 711-720.
5. A. Fiat and M. Naor. Rigorous Time/Space Trade-offs for Inverting Functions. SIAM J. Comput. 29(3): 790-803, 1999.
6. Y. Gertner, S. Kannan, T. Malkin, O. Reingold and M. Viswanathan. The Relationship between Public Key Encryption and Oblivious Transfer. Proc. of the 41st Annual Symposium on Foundations of Computer Science (FOCS), 2000.
7. O. Goldreich. Foundations of Cryptography Basic Tools; Cambridge University Press (2001).
8. O. Goldreich and L. Levin: A Hard-Core Predicate for all One-Way Functions STOC 1989: 25-32.

9. O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR lemma. Technical Report TR95-50, Electronic Colloquium on Computational Complexity, 1995.
10. L. Grover. A Fast Quantum Mechanical Algorithm for Database Search. Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, May 1996, p. 212.
11. I. Haitner, D. Harnik, and O. Reingold. Efficient Pseudorandom Generators from Exponentially Hard One-Way Functions. Proceedings of ICALP 2006, pages 228-239.
12. M. E. Hellman. A Cryptanalytic Time-Memory Trade-Off. IEEE Transactions on Information Theory, Vol. IT-26, N 4, pp. 401–406, 1980.
13. T. Holenstein. Key Agreement from Weak Bit Agreement. Proceedings of STOC 2005, pages 664-673.
14. T. Holenstein. Pseudorandom Generators from One-Way Functions: A Simple Construction for Any Hardness. Proceedings of TCC 2006, pages 443-461.
15. R. Impagliazzo and M. Luby. One-Way Functions are Essential for Complexity-Based Cryptography. 30th IEEE Symposium on Foundations of Computer Science (FOCS), IEEE, pp. 230-235, 1989.
16. R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-way Permutations. Proceedings of the ACM Symposium on Theory of Computing, pages 44-61, 1989.
17. L. A. Levin. One-Way Functions and Pseudorandom Generators. *Combinatorica* 7(4): 357-363, 1987. Earlier version in STOC 1985.
18. U. Maurer. Conditionally-Perfect Secrecy and a Provably Secure Randomized Cipher. Journal of Cryptology, 5(1):53-66, 1992.
19. R. Merkle. Secure communications over insecure channels, CACM April 1978, pages 294-299.
20. O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility between Cryptgraphic Primitives. Proceedings of the First Theory of Cryptography Conference (TCC '04), volume 2951 of Lecture Notes in Computer Science, pages 1-20. Springer-Verlag, 19-21 February 2004.
21. S. Rudich. Limits on the Provable Consequences of One-way Functions. Ph.D. thesis.
22. P.W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantom Computer, SIAM J.Comp., 1997, vol. 26, no. 5, pp. 1484-1509.
23. H. Wee. On obfuscating point functions. STOC 2005: 523-532.