Weak Oblivious Transfer from Strong One-Way Functions

Keisuke Tanaka, Akihiro Yamada, and Kenji Yasunaga

Tokyo Institute of Technology

Abstract. We consider weak oblivious transfer (OT) from strong one-way functions and the paradigm of transforming unconditionally secure protocols in Maurer's bounded storage model into computational secure protocols in the random oracle model. Weak OT is secure against adversaries which have a quadratic resource gap to honest parties. We prove that the random oracle can be replaced with strong one-way functions in the OT protocol. We construct an OT protocol achieving quadratic security from strong one-way functions.

1 Introduction

In modern cryptography, it is important to prove that constructed primitives and protocols are secure. Reduction to computational assumptions is one of the general methods to prove that the primitives and the protocols are secure. For some cryptographic primitives (e.g. private key encryption, pseudo-random generators, bit commitment), several ways of the reduction to weak assumptions are known.

Oblivious transfer (OT) is one of the fundamental cryptographic primitives. OT was first introduced by Rabin [22] and their variations have been studied in several works [24][18][11]. In this paper, we consider the one-out-of-two variant of OT proposed by Even, Goldreich, and Lempel [7], which is shown to be equivalent to Rabin's OT by Crépeau [5] and more useful. The one-out-of-two OT is a protocol between two players: a sender (Alice and a receiver (Bob). Alice has two secrets s_0 and s_1, and Bob has a choice bit c. Bob wishes to receive one out of the two secrets that he chooses (i.e., s_c) without Alice learning c, while Alice wants to ensure that Bob receives only one of the two secrets. OT can be applied to a wide variety of protocols [18][8]. In particular, secure multi-party computation can be based on the security of OT. The constructions of OT require some computational assumptions such as the difficulty of factoring numbers, computing discrete logarithms, and the existence of enhanced trapdoor permutations [22][21][7].

The weakest assumption that is commonly used in cryptography is the existence of one-way functions. If such functions do not exist, any private/public primitives do not exist [15]. Furthermore, the existence of OT cannot be reduced to that of one-way permutations in black-box constructions [16][23]. Therefore, we consider a weaker variant of OT based on one-way functions.

We study the case in which there exists a quadratic gap between the computational resources of honest parties and the adversary. The security achieved in such a setting is called *quadratic security*. In the standard setting, the gap between the computational

resources should be super-polynomial. We here describe the quadratic security using a key-exchange protocol as an example. In a key-exchange protocol, the two honest parties exchange messages and share a common secret key. The standard security guarantees that any adversary that can obtain the secret key needs to compute in super-polynomial time. Since the time complexity of the honest parties is bounded by some polynomial, there is a super-polynomial gap between the computational resources of the honest parties and the adversary. In the quadratic security, if the honest parties can share the secret key in some polynomial time T, then any adversary needs to compute in at least T^2 time to obtain the secret key. Thus, there is a quadratic gap between the computational resources of the honest parties and the adversary rather than a super-polynomial gap.

A key-exchange protocol with quadratic security has been studied by Biham, Goren, and Ishai [2]. They proposed a protocol based on strong one-way functions. Barak and Mahmoody-Ghidary [1] showed that the quadratic gap of the key-exchange protocol in the random oracle model is optimal. Therefore, it is significant to study cryptographic protocols with quadratic security.

1.1 Our Contribution

We present a construction of an OT protocol with quadratic security from a strong one-way function. In our protocol, we use an exponentially strong one-way function. Note that we do not require any trapdoor for one-way functions. The key-exchange protocol presented in [2] also uses such strong one-way functions. Our OT protocol can send an $O(\log k)$-bit secret, where k is the security parameter. We note that our OT protocols only achieve quadratic security for adversary Bob, but they achieve perfect security for adversary Alice. A corresponding situation can make the protocol appealing for situation where Alice is run on a powerful server and Bob is run on a mobile device.

We consider a malicious adversary rather than semi-honest one. In the standard setting, which requires a super-polynomial gap, a malicious OT can be constructed from a semi-honest OT in a black-box way [17][14]. However, a loss of efficiency in the construction is a very sensitive problem in the setting of quadratic security. Therefore, in this paper, we directly construct a malicious OT protocol.

Our protocol is based on Merkle's puzzles [20], the OT protocol in the bounded storage model [6], a standard error-correcting technique, and the paradigm of transforming an unconditionally secure protocol in the bounded storage model into a computationally secure protocol based on strong one-way functions.

A similar transformation was presented for key-exchange protocols [2], and we consider an application of this transformation to OT. First we construct an OT protocol in the random oracle model based on that in the bounded storage model [6]. Second, we replace the random oracle with a strong one-way permutation. Finally, we construct an OT protocol with quadratic security based on strong one-way functions. One problem in replacing a one-way permutation with one-way functions is that one-way functions do not have the 1-to-1 property. Thus, one might consider that we could apply the same technique as in [2] to our protocol. However, it does not work for OT in a simple way. The reason stems from the fact that one of the two parties, Alice and Bob, in the OT protocol can be an adversary. This situation is different from a key-exchange protocol,

in which Alice and Bob are always honest, and the adversary is a third party other than them. In addition, we need to consider a malicious adversary rather than semi-honest one. If we apply the technique in [2] to OT protocols in a straightforward way, then a malicious adversary Bob can obtain both of the two secrets that Alice holds, and this breaks the security of OT. Therefore, in order to circumvent this problem, we use an error-correcting technique to our OT protocol. Due to this technique, our OT protocol achieves the desired functionality and security.

2 Preliminaries

For an integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. We denote by U_n the uniform distribution over $\{0,1\}^n$. For $\ell \leq n$, we write $\binom{[n]}{\ell}$ to denote the set of all subsets $T \subseteq [n]$ with $|T| = \ell$. We write $f(n) = \tilde{O}(g(n))$ if there exists some constant c such that $f(n) = O(g(n)\log^c(g(n)))$. An algorithm is called $T(n)$-*bounded* if the running time on n-bit input is upper bounded by $T(n)$. A function $\epsilon(n)$ is called *negligible* if for any constant $c > 0$, $\epsilon(n) < 1/n^c$ for sufficiently large n. We denote by $\mathrm{negl}(n)$ some negligible function in n. We write $p = \mathrm{poly}(n)$ if $p(\cdot)$ is some polynomial in n. We say that ϵ is *bounded away from* c if $\epsilon(n) \leq c - 1/p(n)$ for some polynomial p for sufficiently large n. We write $x \circ y$ to denote the concatenation operation of x and y.

2.1 Encoding Subsets, Min-Entropy and Strong Extractor

We encode sets into binary strings in the protocols. The following methods are used in [3] and [6]. Using Lemma 1, we can encode $\binom{[n]}{\ell}$ by binary strings of length $\lceil \log \binom{n}{\ell} \rceil$.

Lemma 1 ([4]). *For every integer $\ell \leq n$ there is a 1-to-1 mapping $E : \binom{[n]}{\ell} \to \left[\binom{n}{\ell}\right]$ such that both E and E^{-1} can be computed in time polynomial in n.*

Definition 1 (Dense encoding of subsets). *For every integer $\ell \leq n$ let E be the mapping from Lemma 1. We set $t_m = \lfloor 2^m / \binom{n}{\ell} \rfloor$ where m is an integer such that $m \geq \lceil \log \binom{n}{\ell} \rceil$. We define the mapping $E_m : \binom{[n]}{\ell} \times [t_m] \to \{0,1\}^m$ as $E_m(S, i) = (i-1)\binom{n}{\ell} + E(S)$ (every subset S is mapped to t_m different m bit strings).*

Min-entropy is a variant of Shannon's entropy. It measures the randomness of a random variable or a probability distribution in the worst case. We use an extractor, which is a function that generates uniformly random outputs from high min-entropy distributions. These are also used in [3] and [6]. We review the definitions.

Definition 2 (Min-entropy). *For a distribution X over a probability space Ω, the* min-entropy *of X is defined as*

$$H_\infty(X) = \min_{x \in \Omega} \log(1/\Pr[X = x]).$$

*If $H_\infty(X) \geq k$, X is called k-*source.

Definition 3 (Statistical distance). *Two distributions P and Q over a probability space Ω are* ϵ-close *if for every $A \subseteq \Omega$,*

$$\left| \Pr_{x \leftarrow P}[x \in A] - \Pr_{x \leftarrow Q}[x \in A] \right| \leq \epsilon.$$

We write $P \overset{\epsilon}{\equiv} Q$ if P and Q are ϵ-close. If $\epsilon = 0$, we write $P \equiv Q$.

Definition 4 (Strong extractor). *A function* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* (k, ϵ)-strong extractor *if for every k-source X over $\{0,1\}^n$, the distribution* $(\mathrm{Ext}(X, Y), Y)$ *is ϵ-close to (U_m, Y), where Y is the uniform distribution over $\{0,1\}^d$ and U_m is independent of Y.*

We note that in the standard extractor, which is not strong, the random variables $(\mathrm{Ext}(X, Y), Y)$ and (U_m, Y) are replaced by $\mathrm{Ext}(X, Y)$ and U_m, respectively.

2.2 One-Way Functions and Hard-Core Predicates

Our OT protocol uses strong one-way functions, hard-core predicates, and hard predicates. The following definitions are also used in [2].

Definition 5 (One-way function). *An efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$ is* (T, ϵ) one-way *if for any $T(n)$-bounded adversary A, every sufficiently large n, and the uniform distribution U_n,*

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

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

Definition 6 (Hard-core predicate). *An efficiently computable function $g : \{0,1\}^* \to \{0,1\}$ is a* (T, ϵ) hard-core predicate *for f if for any $T(n)$-bounded adversary A and sufficiently large n,*

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

Definition 7 (Hard predicate). *An efficiently computable function $P : \{0,1\}^* \to \{0,1\}$ is a* (T, ϵ) hard predicate *if for any $T(n)$-bounded adversary A and sufficiently large n,*

$$\Pr_{x \in U_n}[A(f(x)) = P(x)] < \frac{1}{2} + \epsilon(n).$$

Definition 8 (Multi-source hard-core predicate). *An efficiently computable function $H : \{0,1\}^* \to \{0,1\}$ is a* (T, ϵ) multi-source hard-core predicate (MSHCP) *for f if there exist two polynomial $t(\cdot)$ and $s(\cdot)$ such that for any $T(n)$-bounded adversary A and sufficiently large n,*

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

We use Lemmas 2, 3 and 4 to prove the security of our protocols (OWP/OWFs Protocol). Using these lemmas, we can prove that the probability that Bob obtains both of Alice's two secrets is at most $1/\text{poly}(k)$, where k is the security parameter. We can show that this probability is $\text{negl}(k)$ by using dream Yao's XOR lemma (Conjecture 1; see also [2] and [10]). However, we do not know how to prove that this probability is $\text{negl}(k)$ without this conjecture.

Lemma 2 (Goldreich-Levin [9]). *If f is a (T, ϵ) one-way function, then $\text{IP}(x, r) = \langle x, r\rangle$ is a (T', ϵ') hard-core predicate for f with $T'(n) = T(n)\cdot(\epsilon^4/n^3)$ and $\epsilon'(n) = 4\epsilon(n)$, where $\langle x, r\rangle$ is the inner product modulo 2.*

Lemma 3 (Yao's XOR lemma). *If P is a (T, ϵ) hard predicate and it is possible to efficiently sample from distribution $(U_n, P(U_n))$, then for any $\mu(n)$ and $t = \text{poly}(n)$, $P^{(t)}(x_1, \ldots, x_t) = \oplus_{i=1}^{t} P(x_i)$ is a (T', ϵ') hard predicate with $T' = T \cdot (\mu^2/\text{poly}(n)) - \text{poly}(n)$ and $\epsilon' = (2\epsilon)^t + \mu$.*

Conjecture 1 (Dream Yao's XOR lemma [2]). 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 function $\mu(\cdot)$ and some function $\eta(\cdot)$ that is bounded away from 1 such that for any $t = \text{poly}(n)$, $P^{(t)}(x_1, \ldots, x_t) = \oplus_{i=1}^{t} P(x_i)$ is a (T', ϵ') hard predicate with $T' = T \cdot 2^{-o(n)}$ and $\epsilon' = 2^{cn} \cdot \eta^t + \mu(2^n)$.

Lemma 4 ([2]). *For any $\delta < 1$, every $(2^{n(1-\delta)}, \frac{1}{16})$ one-way function has a (T, ϵ) MSHCP with the following T and ϵ:*

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

We note that our OT protocol uses a collection $\mathcal{F}$ of $(2^{n(1-\delta)}, \frac{1}{16})$ one-way functions, and $\mathcal{F}$ can be constructed from a collection of $(2^{n(1-\delta)}, \frac{1}{32})$ one-way functions and an pairwise independent family of hash functions [2].

Definition 9 (Collection of one-way functions). *A (T, ϵ) one-way collection $\mathcal{F} = \bigcup F_n$ of functions is a family of functions $F_n = \{f_i : \{0,1\}^n \to \{0,1\}^* \mid i \in I_n\}$ such that*

Easy to sample and compute: *There exist two PPT algorithms G and F such that $G(1^n)$ outputs an index $i \in I_n$, where $I_n \subseteq \{0,1\}^{\ell(n)}$ and $\ell(n)$ is a polynomial, and for $x \in \{0,1\}^n$, $F(i, x)$ evaluates $f_i(x)$.*

Hard to invert: *For every $T(n)$-bounded adversary A and sufficiently large n,*

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

We say that $\mathcal{F}$ is almost 1-to-1 *if the probability $f_i \in F_n$ is not 1-to-1 is bounded by $2^{-\Omega(n)}$.*

Lemma 5 ([2]). *If there exists a (T, ϵ) one-way function $f : \{0,1\}^{9n} \to \{0,1\}^*$ with $T \geq 2^{\frac{n}{3}}$, $\epsilon \leq 2^{-\frac{n}{3}}$, and $T/\epsilon \geq 2^{9n(1-\delta)}$ then there exists a $(2^{n(1-10\delta)}, \frac{1}{32})$ one-way collection of functions $\mathcal{F} = \bigcup F_n$ that is almost 1-to-1.*

2.3 Oblivious Transfer

We now formally define "weak" oblivious transfer, where "weak" means that malicious player's (either Alice or Bob) computational resources are bounded. First we define malicious strategies for Alice and Bob. In the definition below (Definitions 10 and 11), the computational resources for Alice is weaker than those for Bob.

Definition 10 (Malicious strategy for Alice). *A (malicious) strategy A^* for Alice is an $\tilde{O}(k)$-bounded interactive machine with inputs $s_0, s_1 \in \{0,1\}^{u(k)}$, where k is the security parameter and $u(\cdot)$ is a polynomial. That is, A^* receives s_0, s_1, interacts with B, and in each stage may compute the next message as any function of its inputs, its randomness, and messages it received so far. The view of A^* when interacting with B who holds input c, denoted by $\mathrm{view}_{A^*}^{\langle A^*, B\rangle}(s_0, s_1; c)$, consists of its local output.*

Definition 11 (d-bounded strategy for Bob). *A d-bounded strategy B^* for Bob is an $O(k^{d'})$-bounded interactive machine with input $c \in \{0,1\}$ for any constant $d' < d$. B^* receives c, interacts with A, and in each stage may compute the next message as any function of its inputs, its randomness, and messages it received so far. The view of B^* when interacting with A who holds input s_0 and s_1, denoted by $\mathrm{view}_{B^*}^{\langle A, B^*\rangle}(s_0, s_1; c)$, consists of its local output.*

Definition 12 (C-consistent). *Two pairs $\overline{s} = (s_0, s_1)$ and $\overline{s}' = (s'_0, s'_1)$ are c-consistent if $s_c = s'_c$.*

Definition 13 (Oblivious transfer). *A protocol $\langle A, B\rangle$ is a (d, ϵ)-secure oblivious transfer (OT) protocol if for a security parameter k it is a protocol in which Alice inputs two secrets $s_0, s_1 \in \{0,1\}^{u(k)}$ and Bob inputs a choice bit $c \in \{0,1\}$, and that satisfies:*

Functionality: *If Alice and Bob follow the protocol, then for any s_0, s_1 and c,*

- *The protocol does not abort with probability at least $1 - \mathrm{negl}(k)$.*
- *If the protocol ends then Bob outputs s_c with probability at least $1 - \mathrm{negl}(k)$, whereas Alice outputs nothing.*

Security for Bob: *The view of any strategy A^* for Alice is independent of c. That is, for every s_0 and s_1:*

$$\left\{\mathrm{view}_{A^*}^{\langle A, B^*\rangle}(s_0, s_1; c) \,|\, c = 0\right\} \equiv \left\{\mathrm{view}_{A^*}^{\langle A, B^*\rangle}(s_0, s_1; c) \,|\, c = 1\right\}.$$

(d, ϵ)-Security for Alice: *For every d-bounded strategy B^* for Bob with input c, there is a random variable C defined by the end of the setup stage, which is the stage that Alice does not use her secrets (s_0, s_1) for computation, such that for every two pairs $\overline{s}$ and $\overline{s}'$ that are C-consistent,*

$$\left\{\mathrm{view}_{B^*}^{\langle A, B^*\rangle}(\overline{s}; c)\right\} \overset{\epsilon}{\equiv} \left\{\mathrm{view}_{B^*}^{\langle A, B^*\rangle}(\overline{s}'; c)\right\}.$$

2.4 Interactive Hashing

Our OT protocols use an interactive hashing protocol, in which only Bob has input and both Alice and Bob obtain the same output without Alice learning the input. Below we give the definitions of interactive hashing and describe the protocol which was presented in [6]. The protocol is called 4M-IH Protocol. The work [6] shows that the 4M-IH Protocol satisfies the definitions of interactive hashing (Definitions 15 and 16). We note that the 4M-IH protocol achieves security against unbounded adversaries.

Definition 14 (2^k-to-1 hash functions). *A hash function $h : \{0,1\}^m \to \{0,1\}^{m-k}$ is 2^k-to-1 if for every output of h there are exactly 2^k pre-images. That is, $|h^{-1}(z)| = 2^k$ for every $z \in \{0,1\}^{m-k}$.*

One simple way for constructing a 2^k-to1 hash function is to take a permutation on m-bit strings and omit the last k bits of its output.

Definition 15 (Interactive hashing). *A protocol $\langle A, B\rangle$ is called an* interactive hashing protocol *if it is an efficient protocol between Alice with no input and Bob with input string $W \in \{0,1\}^m$. At the end of the protocol both Alice and Bob output a (succinct representation of a) 2-to-1 function $h : \{0,1\}^m \to \{0,1\}^{m-1}$ and two values $W_0, W_1 \in \{0,1\}^m$ (in the dictionary order) such that $h(W_0) = h(W_1) = h(W)$. Let $e \in \{0,1\}$ be such that $W_e = W$. Furthermore, if the distribution of the string W_{1-e} over the randomness of the two parties is η -close to uniform, then the protocol is called* η-uniform interactive hashing *(or simply uniform interactive hashing if $\eta = 0$).*

Definition 16 (Security of interactive hashing). *An interactive hashing protocol is* secure for B *if for every unbounded deterministic strategy A^*, if h, W_0, W_1 are the outputs of the protocol between an honest Bob with input W and A^*, then*

$$\left\{\text{view}_{A^*}^{\langle A^*,B\rangle}(W) \,|\, W = W_0\right\} \equiv \left\{\text{view}_{A^*}^{\langle A^*,B\rangle}(W) \,|\, W = W_1\right\}.$$

An interactive hashing protocol is (s,ρ)-secure for A *if for every $S \subseteq \{0,1\}^m$ of size at most 2^s, every unbounded deterministic strategy B^*, if h, W_0, W_1 are the outputs of the protocol, then*

$$\Pr[W_0 \in S \cap W_1 \in S] < \rho.$$

where the probability is taken over the randomness of A and B^*.

An interactive hashing protocol is (s,ρ)-secure *if it is secure for B and (s,ρ)-secure for A.*

4M-IH Protocol presented in [6] uses an η-almost t-wise independent permutation constructed in [13]. We describe the protocol below.

Definition 17 (η-almost t-wise independent permutation). *An η-almost t-wise independent permutation is a procedure that takes as input a seed of ℓ bits and outputs a description of an efficiently computable permutation in S_{2^m}, where S_{2^m} is the family of all permutations on m-bit string, with the property that a uniformly chosen seed induces a distribution $\Pi_{t,\eta}$ on permutations such that for any t strings $x_1, \cdots, x_t \in \{0,1\}^m$:*

$$\{\pi(x_1), \cdots, \pi(x_t)\}_{\pi \leftarrow \Pi_{t,\eta}} \overset{\eta}{\equiv} \{\pi(x_1), \cdots, \pi(x_t)\}_{\pi \leftarrow U_{S_m}},$$

where U_{S_m} is the uniform distribution over S_{2^m}.

4M-IH Protocol [6]

Common Input: parameters m and s.
Let $v = s - \log m$.
A family Π of η-almost t-wise independent permutation $\pi : \{0,1\}^m \to \{0,1\}^m$. Set $t = m$ and $\eta = (1/2^v)^t$.
A family G of 2-wise independent 2-1 hash functions $g : \{0,1\}^{m-v} \to \{0,1\}^{m-v-1}$.
A family H of 2-1 hash functions $h : \{0,1\}^m \to \{0,1\}^{m-1}$ defined as:

$$h(x) \stackrel{def}{=} \pi(x)_1 \circ \cdots \circ \pi(x)_v \circ g(\pi(x)_{v+1}, \cdots, \pi(x)_m)$$

where $\pi(x)_i$ denotes the ith bit of $\pi(x)$.
Input of Alice: $\perp$.
Input of Bob: $W \in \{0,1\}^m$

- Alice randomly chooses $\pi \in \Pi$ and sends π to Bob.
- Bob computes $\pi(W) = z_1, \ldots, z_m$ and sends $\pi'(W) = z_1, \ldots, z_v$ to Alice (let π' denote π when truncated in its first v bits).
- Alice randomly chooses $g \in G$ and sends g to Bob.
- Bob sends $g(z_{v+1}, \ldots, z_m)$ to Alice.
- Alice and Bob output W_0, W_1 such that $h(W_0) = h(W_1) = h(W)$.

Lemma 6 ([6]). *For all s and m such that $s \geq \log m + 2$, 4M-IH Protocol is an $(s, 2^{-(m-s)O(\log m)})$-secure η'-uniform interactive hashing protocol for $\eta' = (2^{-(s-\log m-1)})^m < 2^{-m}$. Furthermore, the protocol runs in time polynomial in m.*

Due to lack of space, we omit security proof of Lemma 6 (see [6]).

2.5 Error-Correcting Codes

Our protocol based on one-way functions uses an error-correcting code.

Definition 18. *An error-correcting code* $(\mathrm{Enc}, \mathrm{Dec})$ *consists of two functions* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *and* $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k$ *such that for any $x \in \{0,1\}^k$, we have* $\mathrm{Dec}(\mathrm{Enc}(x)) = x$.

Lemma 7 ([19]). *There exists an error-correcting code* $(\mathrm{Enc}, \mathrm{Dec})$ *such that* $\mathrm{Enc} : \{0,1\}^{3n} \to \{0,1\}^{4n}$ *and* $\mathrm{Dec} : \{0,1\}^{4n} \to \{0,1\}^{3n}$*, and for any $x \in \{0,1\}^{4n}$ we have* $\mathrm{Dec}(\mathrm{Enc}(x) + e) = x$ *for any $e \in \{0,1\}^{4n}$ with Hamming weight less than δn for some constant δ. Furthermore, both* Enc *and* Dec *run in time polynomial in n.*

3 The Protocol in the Random Oracle Model

3.1 Description of ROM Protocol

We describe our OT protocol in the random oracle model (ROM Protocol) below. The protocol uses the following tools, a random permutation, a strong extractor, and an interactive hashing protocol. We assume that k is a power of 2. For arbitrary k, we can think $\log k$ as $\lceil \log k \rceil$ instead. In the following description, the secrets s_0, s_1 are of length $\log k$. We can choose arbitrary $u = \mathrm{poly}(\log k)$ as the length of the secrets. If $s_0, s_1 \in \{0,1\}^u$, we use $\ell = \frac{1}{2} \cdot u^2 \cdot \log^2 k$ and $\mathcal{A}$ and $\mathcal{B}$ of size $k\sqrt{2\ell}$ instead.

ROM Protocol

For security parameter k, we use an oracle to compute a random permutation $f : [k^2] \to [k^2]$ and a random predicate $g : [k^2] \to \{0,1\}$. We set $\ell = \frac{1}{2} \cdot \log^4 k$ and $m = 10\ell \cdot \lceil \log(k \log^2 k) \rceil$.

Input of Alice: Two secrets $s_0, s_1 \in \{0,1\}^{\log k}$.
Input of Bob: A choice bit $c \in \{0,1\}$.
Setup stage:

- Alice chooses a set $\mathcal{A}$ of k^2-bit random strings of size $k \log^2 k$, and queries the oracle on these inputs and sends $f(\mathcal{A}) = \{f(a) \mid a \in \mathcal{A}\}$ to Bob.
- Bob chooses a set $\mathcal{B}$ of k^2-bit random strings of size $k \log^2 k$, and queries the oracle on these inputs to obtain $f(\mathcal{B})$. If $|f(\mathcal{A}) \cap f(\mathcal{B})| < \ell$, then Bob aborts. Otherwise, Bob randomly chooses ℓ common outputs $c_1, \ldots, c_\ell \in f(\mathcal{A}) \cap f(\mathcal{B})$ and sorts $c_1, \ldots, c_\ell$ by the dictionary order.
  Bob randomly chooses $p \in [t_m]$ such that t_m is as in Definition 1 and computes $W = E_m(f(C), p)$, where $f(C) = \{c_1, \ldots, c_\ell\}$.
- Alice and Bob run 4M-IH Protocol, where Bob's input is W. Both Alice and Bob interactively obtain h, W_0, W_1 such that $h(W_0) = h(W_1) = h(W)$
  Alice computes two sets $f(C_0), f(C_1) \subseteq f(\mathcal{A})$ such that $W_0 = E_m(f(C_0), p)$ and $W_1 = E_m(f(C_1), p)$. If W_0 or W_1 is not a valid encoding, i.e., there is no such $f(C_0)$ or $f(C_1)$, then abort.
- Bob chooses e such that $W_e = W$, and sends $\hat{e} = e \oplus c$ to Alice.
- Alice computes $C_0, C_1, g(C_0)$ and $g(C_1)$, where $C_i = \{x \mid f(x) \in f(C_i)\}$ and $g(C_i) = \{g(x) \mid x \in C_i\}$ for $i \in \{0,1\}$, and sorts the elements in the two sets $g(C_0)$ and $g(C_1)$ by the dictionary order.

Transfer stage:

- For $i = 0, 1$, Alice chooses a seed Y_i uniformly at random and computes $Z_i = \mathrm{Ext}(\mathcal{G}(C_i), Y_i) \oplus s_{i \oplus \hat{e}}$, where $\mathcal{G}(C_i)$ is defined as $\mathcal{G}(C_i) = (g_1 \oplus \cdots \oplus g_{\frac{1}{2}\log^2 k}) \circ (g_{\frac{1}{2}\log^2 k+1} \oplus \cdots \oplus g_{\log^2 k}) \circ \cdots \circ (g_{\ell-(\frac{1}{2}\log^2 k-1)} \oplus \cdots \oplus g_\ell)$, where $g_1, \ldots g_\ell \in g(C_i)$. Alice sends to Bob (Y_0, Z_0, Y_1, Z_1).
- Bob computes $C_e = \{x \mid f(x) \in f(C)\}$ and $\mathcal{G}(C_e)$, and then sorts the elements in $\mathcal{G}(C_e)$ by the dictionary order. Then Bob obtains $s_c = \mathrm{Ext}(\mathcal{G}(C_e), Y_e) \oplus Z_e$.

We note that if Alice and Bob follow ROM Protocol, then the protocol requires $\tilde{O}(k)$ running time for both of them. Using an efficient sorting algorithm, they can choose the sets $\mathcal{A}$ and $\mathcal{B}$, sort the sets $f(\mathcal{A}) \cap f(\mathcal{B})$, $g(C_0)$, $g(C_1)$, and $\mathcal{G}(C_e)$ in $\tilde{O}(k)$ time, and query to the random oracles in $O(k)$ time. Alice and Bob can perform the interactive hashing protocol and compute the strong extractor in polynomial time in the input length. Since the input length of the interactive hashing is $|W| = O(m)$ and that of the extractor is $O(|\mathcal{G}(C_i)|) = O(\log^2 k)$, they can perform and compute in $\mathrm{poly}(\log k)$ time. Therefore, they can run ROM Protocol in total $\tilde{O}(k)$ time.

Theorem 1. *Given an oracle to a random function $f : \{0,1\}^* \to \{0,1\}$, there exists a $(2, \epsilon)$ oblivious transfer protocol with $\epsilon = \mathrm{negl}(k)$.*

The proof of this theorem is given in the next section.

3.2 Functionality and Security Proofs

Lemma 8. *ROM Protocol is a* $(2, \epsilon)$ *OT protocol with* $\epsilon = \mathrm{negl}(k)$ *if the extractor is* (k_E, ϵ_E)*-strong for* $k_E \geq \lambda \cdot \log k$ *and* $\epsilon_E = \mathrm{negl}(k)$*, where* λ *is a constant, and the interactive hashing protocol is* (2^{-m})*-uniform and* (ψ, ρ)*-secure for* $\psi \leq m - \ell$ *and* $\rho = \mathrm{negl}(k)$*.*

Lemma 8 follows from Lemmas 9, 10, and 11.

Functionality of ROM Protocol

Lemma 9. *For ROM Protocol that satisfies the parameters in Lemma 8, if Alice and Bob follow the protocol with input* $s_0, s_1 \in \{0,1\}^{\log k}$ *and* $c \in \{0,1\}$*, then the following holds.*

- *The protocol does not abort with probability* $1 - 2^{-\Omega(\ell)}$*;*
- *If the protocol does not abort, then Bob's output is* s_c*.*

Proof. We first show that the protocol does not abort with high probability. The protocol aborts only if $|f(\mathcal{A}) \cap f(\mathcal{B})| < \ell$ or at least one of W_0 and W_1 is not a valid encoding. We show that $|f(\mathcal{A}) \cap f(\mathcal{B})| \geq \ell$ with high probability. For every fixed $f(\mathcal{A})$, since $f(\mathcal{B})$ is a set of random strings, the expected size of $|f(\mathcal{A}) \cap f(\mathcal{B})|$ is $(k \log^2 k)^2 / k^2 \geq \log^4 k = 2\ell$. By Chernoff bound, the probability of $|f(\mathcal{A}) \cap f(\mathcal{B})| < \ell$ is at most $2^{-\Omega(\ell)}$. Next we show that the probability that at least one of W_0 and W_1 is not a valid encoding is small. Since W_e was chosen by Bob, W_e is a valid encoding. By the property of interactive hashing, W_{1-e} is (2^{-m})-close to the uniform distribution over $\{0,1\}^m$. By Lemma 3.3 in [6], the probability that a uniformly random string in $\{0,1\}^m$ is not a valid encoding is at most $|f(\mathcal{A})| \cdot 2^{-m} \leq \binom{k^2}{\ell} \cdot 2^{-m} \leq 2^{-\ell-1}$. Therefore, the probability that the protocol aborts is bounded by $2^{-\Omega(\ell)} + 2^{-m} + 2^{-\ell-1} \leq 2^{-\Omega(\ell)}$.

We show that if the protocol does not abort, then Bob outputs s_c. Since Bob knows C_e, he can always compute $\mathrm{Ext}(g(C_e), Y_e)$ and subsequently use it to decrypt the value Z_e. Then we have

$$\begin{aligned} \mathrm{Ext}(\mathcal{G}(C_e), Y_e) \oplus Z_e &= \mathrm{Ext}(\mathcal{G}(C_e), Y_e) \oplus (s_{e \oplus \hat{e}} \oplus \mathrm{Ext}(\mathcal{G}(C_e), Y_e)) \\ &= s_{e \oplus \hat{e}} \\ &= s_c. \end{aligned}$$

Security for Bob

Lemma 10. *ROM Protocol that satisfies the parameters in Lemma 8 is secure for Bob.*

Proof. We show that the view of any strategy A^* for Alice is independent of c. Fix the randomness of A^*. We show a bijection between pairs of $(\mathcal{B}, c)$ and $(\mathcal{B}', 1-c)$, where $\mathcal{B}$ and $\mathcal{B}'$ are chosen randomly by honest Bob and c is his input. That is, we show that, for each pair $(\mathcal{B}, c)$ that is consistent with the view of A^*, there exists a unique pair $(\mathcal{B}', 1-c)$ such that $(\mathcal{B}', 1-c)$ is consistent with the same view A^*. There are two possibilities for a view of A^*:

– The protocol aborts before the step in which Bob sends the value $\hat{e} = c \oplus e$ to Alice. In this case, the view of A^* is totally independent of c. We can map every consistent $\mathcal{B}$ to itself $\mathcal{B}' = \mathcal{B}$. Clearly $(\mathcal{B}, c)$ and $(\mathcal{B}', 1-c)$ are consistent with the view of A^*.
– Alice receives the message $\hat{e} = c \oplus e$ sent by Bob. Let $(\mathcal{B}, c)$ be a pair consistent with the view of A^*. Then, $f(\mathcal{B})$ is a random set such that $f(C) \subseteq f(\mathcal{A}) \cap f(\mathcal{B})$ and $W_e = E_m(f(C), p)$. Since f is a permutation, $\mathcal{B}$ is also a random set such that $C \subseteq \mathcal{A} \cap \mathcal{B}$. Since the protocol has not aborted, it follows from the property of the interactive hashing that there is a unique random set C' such that $W_{1-e} = E_m(f(C'), p)$. If we set $\mathcal{B}'$ to be $\mathcal{B}' = \mathcal{B} \backslash C \cup C'$, then $(\mathcal{B}', 1-c)$ is consistent with the view of A^*. Therefore we get a bijection.

Since there is a bijection between pairs $(\mathcal{B}, c)$ and $(\mathcal{B}', 1-c)$, the view of A^* for Alice is independent of c.

Security for Alice

Lemma 11. *ROM Protocol that satisfies the parameters in Lemma 8 is* $(2, \epsilon)$*-secure for Alice for* $\epsilon = \mathrm{negl}(k)$.

Proof. Fix a 2-bounded strategy B^* with an input c. We need to show that there exists a random variable C defined by the end of Setup stage such that for every two pairs of secrets $\overline{s} = (s_0, s_1)$ and $\overline{s}' = (s'_0, s'_1)$ that are C-consistent, the view of B^* when the input of Alice is $\overline{s}$ is ϵ-close to that when the input is $\overline{s}'$. In ROM Protocol, the secrets s_0, s_1 are only involved in Transfer stage, and they are sent to Bob as $Z_i = \mathrm{Ext}(\mathcal{G}(C_i), Y_i) \oplus s_i$ for $i \in \{0, 1\}$. Thus, it is sufficient to show that Z_{1-C} is close to the uniform distribution from the view of B^*. To show this, we prove that $\mathcal{G}(C_{1-C})$ has enough entropy from the view of B^*. From the security of the interactive hashing, with probability at least $1-\rho$, B^* obtain no information about $\mathcal{G}(C_{1-C})$ in the interactive hashing. Let assume that such an event happens. Without loss of generality, we assume that B^* makes at most $k^2/3$ different oracle queries. The first bit of $\mathcal{G}(C_{1-C})$ is $g_1 \oplus \cdots \oplus g_{\frac{1}{2}\log^2 k}$. If B^* did not query the oracle for at least one of $g_1, \ldots, g_{\frac{1}{2}\log^2 k}$, then the first bit of $\mathcal{G}(C_{1-C})$ is uniformly distributed, since g_j is an output of a random predicate g. We bound the probability that B^* makes all the elements $g_1, \ldots, g_{\frac{1}{2}\log^2 k}$. Suppose $g_j = g(x_j)$ for $j \in [\frac{1}{2}\log^2 k]$. Let X_j be an event that B^* queries x_j to the oracle. Since g is a random predicate that maps k^2 bits to 1 bit, we have that $\Pr[X_j] \leq 1/k^2$ and thus $\Pr[\bigcup_j X_j] \leq \log^2 k/(2k^2)$. This means that the probability that the first bit of $\mathcal{G}(C_{1-C})$ is not uniformly distributed is at most $\log^2 k/(2k^2)$. The same argument holds for any other bit of $\mathcal{G}(C_{1-C})$. For $i \in [\log k]$, let Y_i be a random variable that takes 1 if an event $\bigcup_j X_{i\cdot(\frac{1}{2}\log^2 k)+j}$ happens and 0 otherwise. Namely, $Y_i = 0$ means that the i-th bit of $\mathcal{G}(C_{1-C})$ is uniformly distributed. Then, the probability that $\mathcal{G}(C_{1-C})$ is a $(\lambda \cdot \log k)$-source is at least $\Pr[\sum_i Y_i \leq (1-\lambda)\log k]$. By Chernoff bound, since $\mathrm{E}[\sum_i Y_i] = \log^2 k \cdot \mathrm{E}[Y_1] \leq \ell/k^2$, we have $\Pr[\sum_i Y_i > (1-\lambda)\log k] \leq 2^{-\Omega(\log^2 k)}$. Thus, with probability at least $1-2^{-\Omega(\log^2 k)}$, $\mathcal{G}(C_{1-C})$ is a $(\lambda \cdot \log k)$-source, and hence $\mathrm{Ext}(\mathcal{G}(C_{1-C}), Y_{1-C})$ is ϵ_E-close to the uniform distribution. Therefore, the statistical distance between the view of B^* when the input of Alice is $\overline{s}$ and that when $\overline{s}'$ is at most $2(\rho + \epsilon_E + 2^{-\Omega(\log^2 k)}) = \mathrm{negl}(k)$.

Replacing Random Permutation with Random Function. We can replace a random permutation with a random function in ROM Protocol using the same technique described in the proof of Theorem 2 in [12]. Thus, Theorem 1 follows from Lemma 8.

4 The Protocol from One-Way Permutations

4.1 Description of OWP Protocol

We construct a protocol with quadratic security based on a one-way permutation. We consider to replace the random oracle with a one-way permutation in ROM Protocol. Note that our protocol relies on a (T, ϵ) one-way permutation with $T(n) \geq 2^{\log k}$. We assume that k is a power of 2. For simplicity, in the following description, the secrets s_0, s_1 are of length $\log k$. For arbitrary $u = \mathrm{poly}(\log k)$, we use $s_0, s_1 \in \{0,1\}^u$ and $\ell = t \cdot u$ instead.

OWP Protocol

For a security parameter k, we use a one-way permutation $f : \{0,1\}^{2\log k} \to \{0,1\}^{2\log k}$ for which H is a (T, ϵ_H) MSHCP with $T = 2^{2(1-\delta)\cdot\log k}$ for a positive constant $\delta < 1$. We set $\ell = t \cdot \log^2 k$ and $m = 10\ell \cdot \lceil \log(k\sqrt{2\ell}) \rceil$, where $t = t(\log k)$ is some polynomial for H as in Definition 8. Also let $s = s(\log k)$ be some polynomial for H as in Definition 8.

Input of Alice: Two secrets $s_0, s_1 \in \{0,1\}^{\log k}$.
Input of Bob: A choice bit $c \in \{0,1\}$.
Setup stage:

- Alice chooses a set $\mathcal{A}$ of $2\log k$-bit random strings of size $\lceil k\sqrt{2\ell} \rceil$ and a set $\mathcal{R}_i$ of s-bit random strings of size $\log^2 k$ for $i = 0, 1$, and computes $f(\mathcal{A}) = \{f(a) \mid a \in \mathcal{A}\}$ and sends them to Bob.
- Bob chooses a set $\mathcal{B}$ of $2\log k$-bit random strings of size $\lceil k\sqrt{2\ell} \rceil$, and computes $f(\mathcal{B}) = \{f(b) \mid b \in \mathcal{B}\}$. If $|f(\mathcal{A}) \cap f(\mathcal{B})| < \ell$, then Bob aborts. Otherwise, Bob randomly chooses ℓ common outputs $c_1, \ldots, c_\ell \in f(\mathcal{A}) \cap f(\mathcal{B})$.
  Bob randomly chooses $p \in [t_m]$ such that t_m is as in Definition 1 and computes $W = E_m(f(C), p)$, where $f(C) = \{c_1, \ldots, c_\ell\}$.
- Alice and Bob run 4M-IH Protocol, where Bob's input is W. Both Alice and Bob interactively obtain h, W_0, W_1 such that $h(W_0) = h(W_1) = h(W)$.
  Alice computes two sets $f(C_0), f(C_1) \subseteq f(\mathcal{A})$ such that $W_0 = E_m(f(C_0), p)$ and $W_1 = E_m(f(C_1), p)$. If W_0 or W_1 is not a valid encoding, then abort.
- Bob chooses e such that $W_e = W$, and sends $\hat{e} = e \oplus c$ to Alice.
- For $i = 0, 1$, Alice computes C_i and $\mathcal{H}(C_i, \mathcal{R}_i)$, where $C_i = \{x \mid f(x) \in f(C_i)\}$ and $\mathcal{H}(C_i, \mathcal{R}_i) = H(x_1, \ldots, x_t, r_1) \circ H(x_{t+1}, \ldots, x_{2t}, r_2) \circ \cdots \circ H(x_{\ell-t+1}), \ldots, x_\ell, r_{\log^2 k})$, where $x_1, \ldots x_\ell \in C_i, r_1, \ldots r_{\log^2 k} \in \mathcal{R}_i$. Alice sorts the elements in $\mathcal{H}(C_0, \mathcal{R}_0)$ and $\mathcal{H}(C_1, \mathcal{R}_1)$ by the dictionary order.

Transfer stage:

- For $i = 0, 1$, Alice chooses a seed Y_i uniformly at random and computes $Z_i = \mathrm{Ext}(\mathcal{H}(C_i, r_i), Y_i) \oplus s_{i\oplus\hat{e}}$. Alice sends to Bob (Y_0, Z_0, Y_1, Z_1).

– Bob computes $C_e = \{x \mid f(x) \in f(C)\}$ and $\mathcal{H}(C_e, r_e)$, and then sorts the elements in $\mathcal{H}(C_e, r_e)$ by the dictionary order. Then Bob obtains $s_c = \mathrm{Ext}(\mathcal{H}(C_e, r_e), Y_e) \oplus Z_i$.

We note that if Alice and Bob follow OWP Protocol, then they can run it in total $\tilde{O}(k)$ time. This follows from a similar argument as ROM Protocol.

Theorem 2. *If there exists a* $(2^{2\log k(1-\delta)}, \frac{1}{16})$ *one-way permutation for* $\delta < \frac{1}{2}$*, then there exists a* (d, ϵ)*-secure oblivious transfer protocol for the following* d *and* ϵ*:*

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

4.2 Functionality and Security Proofs

We first show the following lemma.

Lemma 12. *OWP Protocol is a* $(2, \epsilon)$ *OT protocol with* $\epsilon = 2\epsilon_H \cdot \log^2 k + \mathrm{negl}(k)$ *if* H *is* (T, ϵ_H) *MSHCP, the extractor is* (k_E, ϵ_E)*-strong for* $k_E \geq \lambda \cdot \log^2 k$ *and* $\epsilon_E = \mathrm{negl}(k)$*, where* λ *is a constant, and the interactive hashing protocol is* (2^{-m})*-uniform and* (ψ, ρ)*-secure for* $\psi \leq m - \ell, \rho = \mathrm{negl}(k)$*.*

Theorem 2 can be proven by this lemma and Lemma 4. Lemma 12 follows from Lemmas 13, 14, and 15.

Functionality of OWP Protocol

Lemma 13. *For OWP Protocol that satisfies in Lemma 12, if Alice and Bob follow the protocol with input* $s_0, s_1 \in \{0,1\}^{\log k}$ *and* $c \in \{0,1\}$*, then the following holds.*

– *The protocol does not abort with probability* $1 - 2^{-\Omega(\ell)}$*;*
– *If the protocol does not abort, then Bob's output is* s_c*.*

We can show this lemma by almost the same argument as the proof of Lemma 10. Hence we omit the proof.

Security for Bob

Lemma 14. *OWP Protocol that satisfies the parameters in Lemma 12 is secure for Bob.*

Since the proof is almost the same as Lemma 9, we omit the proof.

Security for Alice

Lemma 15. *OWP Protocol that satisfies the parameters in Lemma 12 is* $(2, \epsilon)$*-secure for Alice with* $\epsilon = 2\epsilon_H \cdot \log^2 k + \mathrm{negl}(k)$.

Proof. Fix a 2-bounded strategy B^* with an input c. We need to show that there exists a random variable C defined by the end of Setup stage such that for every two pairs of secrets $\overline{s} = (s_0, s_1)$ and $\overline{s}' = (s'_0, s'_1)$ that are C-consistent, the view of B^* when the input of Alice is $\overline{s}$ is ϵ-close to that when the input is $\overline{s}'$. By the same reason as in Lemma 11, it is sufficient to show that $\mathcal{H}(C_{1-C}, \mathcal{R}_{1-C})$ has enough entropy from the view of B^*. In the case of ROM Protocol (Lemma 11), we show that $\mathcal{G}(C_{1-C})$ has enough entropy. The difference is that, in OWP Protocol, $\mathcal{H}(C_{1-C}, \mathcal{R}_{1-C})$ consists of the outputs of MSHCP, while in ROM Protocol, $\mathcal{G}(C_{1-C})$ consists of the outputs of the random predicate g. Since H is (T, ϵ_H) MSHCP and $|\mathcal{H}(C_{1-C}, \mathcal{R}_{1-C})| = \log^2 k$, the difference is at most $\epsilon_H \cdot \log^2 k$. Therefore, by the same argument as in the proof of Lemma 11, the statistical distance between the view of B^* when the input of Alice is $\overline{s}$ and that when $\overline{s}'$ is at most $2(\rho + \epsilon_E + 2^{-\Omega(\log^2 k)} + \epsilon_H \cdot \log^2 k) \leq \epsilon$.

5 The Protocol from One-Way Functions

5.1 Description of OWFs Protocol

We describe our OT protocol based on one-way functions (OWFs Protocol) below. The protocol uses the following tools, a collection of one-way functions, an interactive hashing protocol, a strong extractor, and an error-correcting code.

In OWFs Protocol, Alice first randomly chooses two random strings σ_0 and σ_1, which are the same length as s_0 and s_1. Then Alice encodes σ_0, σ_1 to $\sigma'_0\sigma'_1$ using an error-correcting code. Alice and Bob run the basic protocol several times. In the one basic protocol, they run setup stage several times. Bob's output of the t'th basic protocol is the t'th bit of the encoded string σ'_c. If Bob collects almost of all outputs of the basic protocols, then Bob can decode the strings σ'_c to σ_c which Alice made, and by using it, get Alice's input s_c.

For simplicity, we assume that k is a power of 2. In OWFs Protocol, for simple description, we set the length of the secrets (s_0 and s_1) with $\log^2 k$ while that of the previous ROM/OWP Protocol with $\log k$. We can choose arbitrary $u = \mathrm{poly}(\log k)$ as the length of the secrets.

OWFs Protocol

For a security parameter k, we use a collection $\mathcal{F} = \bigcup F_{(2\log k)}$ of one-way functions, 4M-IH Protocol with input length $m = 10\ell \cdot \lceil \log k \sqrt{2\ell} \rceil$, a randomness extractor, and an error-correcting code $(\mathrm{Enc}, \mathrm{Dec})$ with $\mathrm{Enc} : \{0,1\}^u \to \{0,1\}^{u'}$, where $\ell = \frac{1}{2} \cdot \log^2 k$, $u = \log^2 k$, and $u' = \frac{4}{3}u$.

Input of Alice: Two secrets $s_0, s_1 \in \{0,1\}^u$.
Input of Bob: A choice bit $c \in \{0,1\}$.

Encoding stage:

- Alice chooses two random strings $\sigma_0, \sigma_1 \in \{0,1\}^u$, and computes $\sigma'_0 = \mathrm{Enc}(\sigma_0)$ and $\sigma'_1 = \mathrm{Enc}(\sigma_1)$.

For each $t \in [u']$, Alice and Bob run the following basic protocol.

The basic protocol for t :

Alice and Bob run Setup stage for each $s \in [\log k]$, and then run Combining stage.

Setup stage for s :

- Alice chooses a set $\mathcal{A}$ of $2\log k$-bit random strings of size $k\log k$, two random strings $r_0, r_1 \in \{0,1\}^\ell$, and $f_i \in F_{(2\log k)}$. Alice computes $f_i(\mathcal{A}) = \{f_i(x) \mid x \in \mathcal{A}\}$ and sends it to Bob.
- Bob chooses a set $\mathcal{B}$ of $2\log k$-bit random strings of size $k\log k$, and computes $f_i(\mathcal{B}) = \{f_i(x) \mid x \in \mathcal{B}\}$. If $|f_i(\mathcal{A}) \cap f_i(\mathcal{B})| < \ell$, then Bob aborts. Otherwise, Bob randomly chooses ℓ common outputs $c_1, \ldots, c_\ell \in f(\mathcal{A}) \cap f(\mathcal{B})$. Bob randomly chooses $p \in [t_m]$ such that t_m is as in Definition 1 and computes $W = E_m(f_i(C), p)$, where $f_i(C) = \{c_1, \ldots, c_\ell\}$.
- Alice and Bob run the 4M-IH Protocol, where Bob's input is W. Both Alice and Bob interactively obtain h, W_0, W_1 such that $h(W_0) = h(W_1) = h(W)$. Alice computes two sets $f(C_0), f(C_1) \subseteq f(\mathcal{A})$ such that $W_0 = E_m(f(C_0), p)$ and $W_1 = E_m(f(C_1), p)$. If W_0 or W_1 is not a valid encoding, then abort.
- Bob chooses e such that $W_e = W$, and sends $\hat{e}_s = e \oplus c$ to Alice.
- Alice computes C_0, C_1 and sorts the elements by the dictionary order, where $C_j = \{x \mid f_i(x) \in f(C_j)\}$ for $j \in \{0,1\}$, Alice randomly chooses two indices $i_0, i_1 \in [\ell]$ such that $c_{i_0} \in C_0 \setminus C_1$ and $c_{i_1} \in C_1 \setminus C_0$, computes $v^s_0 = \mathrm{IP}(c_{i_0}, r_0)$ and $v^s_1 = \mathrm{IP}(c_{i_1}, r_1)$ and sends (i_0, i_1) to Bob.
- Bob computes $v^s_{\hat{e}_s} = \mathrm{IP}(c_{\hat{e}_s}, r_{\hat{e}_s})$.

Combining stage:

- Alice computes $V^t_0 = v^1_{\hat{e}_1} \oplus v^2_{\hat{e}_2} \oplus \cdots \oplus v^{\log k}_{\hat{e}_{\log k}}$, $V^t_1 = v^1_{(1-\hat{e}_1)} \oplus v^2_{(1-\hat{e}_2)} \oplus \cdots \oplus v^{\log k}_{(1-\hat{e}_{\log k})}$.
- Bob computes $V^t_c = v^1_c \oplus v^2_c \oplus \cdots \oplus v^{\log k}_c$.

Transfer stage:

- For $i = 0, 1$, Alice computes $V_i = (V^1_i \circ V^2_i \circ \cdots \circ V^{u'}_i)$ and $V'_i = V_i \oplus \sigma'_i$, and chooses a random seed Y_i and computes $Z_i = \mathrm{Ext}(\sigma_i, Y_i) \oplus s_i$. Then Alice sends (V'_0, Z_0, V'_1, Z_1) to Bob.
- Bob computes $\hat{\sigma}'_c = V'_c \oplus (V^1_c \circ V^2_c \circ \cdots \circ V^{\log k}_c)$ and $\sigma_c = \mathrm{Dec}(\hat{\sigma}'_c)$, and obtains $s_c = \mathrm{Ext}(\sigma_c, Y_c) \oplus Z_c$.

We note that if Alice and Bob follow OWFs Protocol, then the protocol requires $\tilde{O}(k)$ running time for them. In Encoding stage, Alice can choose strings and encode them in $\mathrm{poly}(u) = \mathrm{poly}(\log k)$ time. In Setup stage for each Basic protocol, Alice and Bob can choose the sets $\mathcal{A}$ and $\mathcal{B}$, sort the sets $f_i(\mathcal{A}) \cap f_i(\mathcal{B})$, choose elements i_0 and i_1 , and compute v^s_0, v^s_1 and $v^s_{\hat{e}_s}$ in $\tilde{O}(k)$. Alice and Bob can perform 4M-IH Protocol in polynomial time in the input length, that is, $\mathrm{poly}(\log k)$ time. Since, they run Setup stage $s = \log k$ times and compute V^t_0, Z^t_1 and Z^t_c in Combining stage in $O(s) = O(\log k)$, they perform the basic protocol in total $\tilde{O}(k)$ time. They can perform Transfer stage in $\mathrm{poly}(\log k)$ time. Therefore, they can run OWFs Protocol in total $\tilde{O}(k)$ time.

Theorem 3. *For any $\delta < 1/10$, If there exists a (T, ϵ) one-way functions with $T \geq 2^{2\log k/3}$, $\epsilon \leq 2^{-(2\log k/3)}$, and $T/\epsilon \geq 2^{18\log k(1-\delta)}$, then there exists a (d, ϵ)-secure oblivious transfer protocol for the following d and ϵ:*

- $d = 2 \cdot (1 - 10\delta)$ $\epsilon = 1/\log^c k$ for any constant c;
- $d = 2 \cdot (1 - 10\delta - \tau)$ $\epsilon = k^{-\tau}$ for any $\tau < 1 - \delta$;
- $d = 2 \cdot (1 - 10\delta)$ $\epsilon = \mathrm{negl}(k)$ assuming the dream XOR lemma.

5.2 Functionality and Security Proofs

We first show the following lemma:

Lemma 16. *OWFs Protocol is a* $(2, \epsilon)$ *OT protocol with* $\epsilon = \epsilon_F \cdot \mathrm{poly}(\log k) + \mathrm{negl}(k)$ *if* $\mathcal{F}$ *is a* (T, ϵ_F) *one-way collection of function that is almost 1-to-1, the extractor is* (k_E, ϵ_E)*-strong for* $k_E \geq \lambda \cdot \log^2 k$ *and* $\epsilon_E = \mathrm{negl}(k)$*, where* λ *is a constant, the interactive hashing protocol is* (2^{-m})*-uniform and* (ψ, ρ)*-secure for* $\psi \leq m - \ell, \rho = \mathrm{negl}(k)$*, and the error-correcting code is as presented in Lemma 7.*

Theorem 3 can be proven by this lemma and Lemma 5. Lemma 16 follows from Lemmas 17, 18, 19.

Functionality of OWFs protocol

Lemma 17. *For OWFs Protocol that satisfies in Lemma 16, if Alice and Bob follow the protocol with input* $s_0, s_1 \in \{0,1\}^{\log k}$ *and* $c \in \{0,1\}$*, then the following holds.*

- *The protocol does not abort with probability* $1 - 2^{-\Omega(\ell)}$*;*
- *If the protocol does not abort, then Bob's output is* s_c *with probability* $1 - \mathrm{negl}(k)$.

Proof. The protocol aborts only in running Setup stage of the basic protocol. Since OWFs Protocol performs Setup stage $\mathrm{poly}(\log k)$ times, it follows from the same argument as in the case of ROM/OWP Protocol that the probability that the protocol aborts is $\mathrm{negl}(k)$.

Next we show that, if the protocol does not abort, the output of Bob is s_c with probability at least $1 - \mathrm{negl}(k)$. Bob can output s_c if he can compute $\mathrm{Ext}(\sigma_c, Y_c)$ correctly, which happens if $\mathrm{Dec}(\hat{\sigma}'_c) = \sigma'_c$. For each $t \in [u']$, the probability that the value of V_c^t that Bob holds differs from that of Alice is at most $(\log k)/k^2$. Thus, the expected number of errors contained in $\hat{\sigma}'_c$ is $(\log^3 k)/k^2$. Since Dec can correct $\Omega(\log^2 k)$ errors in $\hat{\sigma}'_c$, we can show by using Chernoff bound that the probability that $\mathrm{Dec}(\hat{\sigma}'_c) = \sigma'_c$ is at least $1 - \mathrm{negl}(k)$.

Security for Bob

Lemma 18. *OWFs Protocol that satisfies the parameters in Lemma 16 is secure for Bob.*

Proof. Bob uses his secret c only in the iteration of Setup stage. By the same argument as in ROM/OWP Protocol, Alice has no information about c in the information theoretic sense. Therefore, the lemma follows.

Security for Alice

Lemma 19. *OWFs Protocol that satisfies the parameters in Lemma 16 is* $(2, \epsilon)$*-secure for Alice with* $\epsilon = 2\epsilon_F \cdot \mathrm{poly}(\log k) + \mathrm{negl}(k)$

Proof. For a 2-bounded strategy B^* with an input c, we need to show that there exists a random variable C defined by the end of Setup stage such that for every two pairs of secrets $\overline{s} = (s_0, s_1)$ and $\overline{s}' = (s'_0, s'_1)$ that are C-consistent, the view of B^* when the input of Alice is $\overline{s}$ is ϵ-close to that when the input is $\overline{s}'$. As in the case of ROM/OWP Protocol, it is sufficient to show that the input to the extractor, σ_{1-C}, has enough entropy.

First we show that the probability that Bob can obtain some bit of both σ'_C and σ'_{1-C} is small. To obtain a bit of both σ'_C and σ'_{1-C}, he need to break either the interactive hashing or the one-wayness of f_i for all $\log k$ iterations of Setup stage. That probability is at most $(\frac{1}{2} + \epsilon_F + \rho)^{\mathrm{poly}(\log k)} \leq \epsilon_F \cdot \mathrm{poly}(\log k)$. Therefore, the probability that Bob obtains some bit of both σ'_C and σ'_{1-C} is at most $\epsilon_F \cdot \mathrm{poly}(\log k)$.

Thus, if Bob obtains σ'_C correctly, then he performs the basic protocol with input C at least $\frac{1}{2}u'$ times out of u' times. This means that he performs the basic protocol with input $1 - C$ at most $\frac{1}{2}u'$ times. By a similar argument as is the proof of Lemma 11, σ_{1-C} has entropy $\geq \frac{3}{4}u$. Hence the entropy of σ'_C is at least $\frac{3}{4}u - \frac{1}{2}u' = \frac{1}{12}u = \Omega(\log^2 k)$.

Therefore, the statistical distance between the view of B^* when the input of Alice is $\overline{s}$ and that when $\overline{s}'$ is at most $2(\epsilon_F \cdot \mathrm{poly}(\log k) + \epsilon_E) \leq \epsilon$.

6 Conclusions and Open Problems

We have proposed an OT protocol with quadratic security from strong one-way functions. In our OT protocol (OWFs Protocol), Bob obtains the two secret with probability $1/\mathrm{poly}(k)$. We do not know the way of improving this probability to $\mathrm{negl}(k)$ without dream Yao's XOR lemma. In order to do this, we believe that we must use other techniques: other error-correcting codes, extractors. Thus, these techniques may improve probability $\mathrm{negl}(k)$ and remain hardness.

Another issue is on optimality. In [1], it is shown that the key-exchange protocol with quadratic security is optimal in the random oracle model. Therefore, it may be able to prove that an OT protocol with quadratic security is optimal in the random oracle model.

Acknowledgement

We are very grateful to Dr. Claudio Orlandi for helpful suggestions and comments. We would also like to thank the anonymous referees for very valuable comments.

References

1. B. Barak and M. Mahmoody-Ghidary. Merkle Puzzles Are Optimal – An $O(n^2)$-query Attack on Any Key Exchange from a Random Oracle. *Proceedings of* CRYPTO 2009, pages 374–390, 2009.

2. E. Biham, Y. J. Goren, and Y. Ishai. Basing Weak Public-Key Cryptography on Strong One-Way Functions. *Proceedings of* TCC 2008, pages 55–72, 2008.
3. C. Cachin, C. Crépeau, and J. Marcil. Oblivious Transfer with a Memory-Bounded Receiver. *Proceedings of* FOCS 1998, pages 493–502, 1998.
4. T. M. Cover. Enumerative Source Encoding. *IEEE Transaction on Information Theory*, pages 73–77, 1973.
5. C. Crépeau. Equivalence between Two Flavours of Oblivious Transfers. *Proceedings of* CRYPTO 1987, pages 350–354, 1987.
6. Y. Z. Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-Round Oblivious Transfer in the Bounded Storage Model. *Proceedings of* TCC 2004, pages 446–472, 2004.
7. S. Even, O. Goldreich, and A. Lempel. A Randomized Protocol for Signing Contracts. *Communications of the ACM*, pages 637–647, 1985.
8. Y. Gertner, S. Kannan, and T. Malkin. The Relationship between Public Key Encryption and Oblivious Transfer. *Proceedings of* FCCS 2000, pages 325–335, 2000.
9. O. Goldreich. *Foundation of Cryptography Basic Tools*. Cambridge University Press, 2001.
10. O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR lemma. *Technical Report TR95-50*, 1995.
11. S. Goldreich, O. Micali and A. Wigderson. How to Play Any Mental Game - A Completeness Theorem for Protocols with Honest Majority. *In 19th ACM Symposium on the Theory of Computing*, pages 218–229, 1987.
12. Y. Goren. Basing Weak Public-Key Cryptography on Strong One-Way Functions. *M.Sc. Thesis, Technion*, 2006.
13. W. T. Gowers. An Almost m-wise Independent Random Permutation of the Cube. *Combinatorics, Probability and Computing*, pages 119–130, 1996.
14. I. Haitner. Semi-honest to Malicious Oblivious Transfer - The Black-Box Way. *Proceedings of* TCC 2008, pages 412–426, 2008.
15. R. Impagliazzo and M. Ruby. One-Way Functions Are Essential for Complexity-Based Cryptography. *Proceedings of* FOCS 1989, pages 230–235, 1989.
16. R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-Way Permutations. *Proceedings of* STOC 1989, pages 44–61, 1989.
17. Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank. Black-Box Constructions for Secure Computation. *Proceedings of* STOC 2006, pages 531–540, 2006.
18. J. Kilian. Founding Cryptography on Oblivious Transfer. *In 20th ACM Symposium on the Theory of Computing*, pages 20–31, 1988.
19. F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1977.
20. R. C. Merkle. Secure Communications over Insecure Channels. *Communications of the ACM*, pages 294–299, 1978.
21. M. Naor and B. Pinkas. Efficient Oblivious Transfer Protocols. *Proceedings of* SODA 2001, pages 448–457, 2001.
22. M. Rabin. How to Exchange Secrets by Oblivious Transfer. *Technical Report TR-81*, 1981.
23. O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of Reducibility between Cryptographic Primitives. *Proceedings of* TCC 2004, pages 1–20, 2004.
24. A. C. Yao. How to Generate and Exchange Secrets. *In 27th IEEE Symposium on Theory of Computing*, pages 218–229, 1986.