Computational Oblivious Transfer and Interactive Hashing

Kirill Morozov* and George Savvides†

May 28, 2009

Abstract

We use interactive hashing to achieve the most efficient OT protocol to date based solely on the assumption that trapdoor permutations (TDP) exist. Our protocol can be seen as the following (simple) modification of either of the two famous OT constructions: 1) In the one by Even et al (1985), a receiver must send a random domain element to a sender through IH; 2) In the one by Ostrovsky et al (1993), the players should use TDP instead of one-way permutation. A similar approach is employed to achieve oblivious transfer based on the security of the McEliece cryptosystem. In this second protocol, the receiver inputs a public key into IH, while privately keeping the corresponding secret key. Two different versions of IH are used: the computationally secure one in the first protocol, and the information-theoretically secure one in the second.

Keywords: Oblivious transfer, interactive hashing, trapdoor permutation, McEliece cryptosystem

1 Introduction

Oblivious Transfer. It is one of the central cryptographic primitives, since it implies secure two- (and multi-)party computation [11]. Oblivious transfer (OT) was initially proposed in several flavors [29, 26, 8] which all turned out to be equivalent [4]. We will focus on the *one-out-of-two (1-2) OT* [8] which is a two-party primitive, where a honest sender inputs two bits b_0 and b_1 and a honest receiver chooses to obtain one of them, the *chosen bit* b_c, by inputting his *choice bit* c. The cheating sender must remain ignorant of c, while the cheating receiver – unable to learn both bits.

Several frameworks for constructing computationally secure[1] OT are known, in particular, using: generic assumptions [8, 12], smooth projective hashing [17], lossy trapdoor functions [25], dual-mode cryptosystems [24]. Each of them enjoys different advantages, such as generality of underlying assumptions, efficiency, advanced security properties (e.g. composability).

Interactive Hashing. This primitive has first appeared as a sub-routine in the oblivious transfer protocol with computationally unbounded party [23]. Later, interactive hashing (IH) has proven to have many other applications in cryptography, for instance, in bit commitment, zero-knowledge, and unconditional oblivious transfer protocol design (see e.g. [27, Sec. 2.1] for a survey). IH is a two-party primitive, where a honest sender inputs a string $w \in \{0,1\}^n$, and both it and a honest receiver obtain on the output (w, w'), where $w' \in_R \{0,1\}^n \backslash w$. The *hiding* property requires that the cheating receiver cannot tell which of (w, w') was the input. According to the *binding* property, at least one of (w, w') is effectively beyond the control of the cheating sender. IH comes in two versions: with computational

*RCIS, AIST, Japan. E-mail: kirill.morozov@aist.go.jp

†Independent researcher, Germany. E-mail: gsavvides@gmail.com

[1]That is OT based on computational assumptions, where at least one player is computationally bounded.

binding (initiated by [23]), we call it C-IH, and information-theoretic binding (initiated by [2]), we call it IT-IH. Hiding is information-theoretic in both flavors.

Related Work. In fact, C-IH has already been used for implementing OT from one-way permutations (OWP) and one-way functions (OWF) [23], but in that work, one of the honest players had to invert OWP/OWF (i.e. to be computationally strong). In contrast, our work admits honest players which are bounded to probabilistic polynomial time (PPT) – for the price of strengthening the assumption to TDP.

OT from TDP is considered in a line of works [8, 12], where the protocols are designed to be secure against passively cheating players. Protection against active attackers is achieved using secure compilers [11, 13]. Our TDP-based protocol can be considered complementary to [13], where a more general case of "sparse" domain encodings is considered. Our construction shows that in a particular case, when the TDP's domain can be succinctly represented by binary strings (i.e. all the encodings are valid),[2] C-IH can substitute a secure compiler.

OT from coding based assumptions was recently constructed in [6]. This scheme is more efficient than our coding based protocol, whose bottleneck is the (round- and communication-)efficiency of a currently available IT-IH protocol (see Appendix A). Moreover, they do not use any non-standard assumptions, compared to our case. On the other hand, our scheme does not employ commitments as they do. Furthermore, up to date, there is no lower bound on efficiency of IT-IH (in contrast to C-IH), hence any improvements to this end may apply to our scheme as well.

A construction in the spirit of our coding based scheme (Protocol 2) has been independently discovered by Claude Crépeau and Jörn Müller Quade [20].

Our contribution. We present a novel connection between the primitives of oblivious transfer and interactive hashing by constructing two (quite simple) protocols using the two versions of IH and, correspondingly, two types of computational assumptions.

Our first protocol can be considered as the following (simple) modification of either of the two famous OT constructions: 1) In [8], a receiver must send a random domain element to a sender using C-IH; 2) In [23], the players should use TDP instead of one-way permutation. Our protocol requires one instance of C-IH plus one additional pass. Hereby, we achieve the round-optimal OT protocol with information-theoretic receiver-security based on TDP in blackbox manner. Its round complexity matches the linear lower bound presented in [15] (see also [14, Sec. 7.3]).

The second protocol extends our approach by constructing OT (with computational security for both players) using the coding based assumptions underlying security of the McEliece public key cryptosystem (PKC) [19]. This protocol requires one instance of IT-IH plus one additional pass. Here, a receiver sends a public key to a sender using IT-IH, while privately keeping the corresponding secret key. In order to simplify the proof of sender-security, we take a non-standard coding assumption: loosely speaking, we suppose that there are quite a few codes which (similarly to the irreducible Goppa codes) can work as public keys for the McEliece-type encryption.

Our constructions manifest importance of the IH primitive in the following sense: if IH is available as a blackbox, then under appropriate assumptions, OT can be achieved using only two passes. In some sense, IH works as a "security compiler" denying certain malicious player's behavior.

Organization. Section 2 contains a description of basic notation and tools. The TDP-based protocol using C-IH is presented in Sec. 3, while the coding based protocol using IT-IH is presented in Sec. 4. Sec. 5 discusses possible extensions and open questions.

2 Preliminaries

Basic Notation. Summation is bitwise exclusive-or. By *weight*, we refer to the Hamming weight. $\mathbf{I}$ is the unit matrix. "$[M_0|M_1]$" denotes a concatenation of matrices M_0, M_1 of

[2]Note that this assumption is implicitly present in [23, 21].

appropriate size.

An element x uniformly distributed in the domain X is denoted by "$x \in_R X$". By writing that x is *negligible* in n, we mean that x is decreasing faster than any polynomial fraction in a security parameter n. Whenever mentioning of the security parameter is omitted, we implicitly refer to n (whatever it denotes in the given context). When the statement is claimed to hold *on the average*, it holds for all but a negligible fraction of instances. We call an algorithm *efficient*, if it is PPT. Computational indistinguishability is denoted by "$\stackrel{c}{=}$".

A *view* of a player participating in an interactive protocol represents the player's inputs, results of all local computations and local coin tosses, and messages exchanged. The view of a player A having input x and interacting with a player B having input y is denoted by $View_{\mathsf{A}}(\mathsf{A}(x), \mathsf{B}(y))$.

Speaking of *information-theoretic* (or *unconditional*) security, we refer to protection against computationally unbounded adversary. In this case, a security failure probability, negligibly small in some security parameter, is admitted.

Adversary Model. We consider the *static* adversary, i.e. either a sender or a receiver gets corrupt prior to the protocol execution. A player is called *honest*, if it strictly follows the protocol. A *passive* attacker follows the protocol, but may use his view. Finally, an *active attacker*, in addition to the above, may deviate from the protocol arbitrarily.

Linear codes. A binary linear (n, k) code of *length* n and *dimension* k is a k-dimensional linear subspace of $\{0,1\}^n$. In particular, it can be represented by a rank-k *generating matrix* $G \in \{0,1\}^{k \times n}$ (i.e. its rows are linearly independent). Any such G can be equivalently (up to a column permutation) represented in the standard form: $[\mathbf{I}^k | A]$ for some $A \in \{0,1\}^{k \times n-k}$. Let us call A the *standard part* of such G. Throughout this paper, we consider representation of (n, k) linear codes by their standard parts, denoting the set of all possible matrices A by $\mathcal{C}$.

For further information on linear codes, we refer the reader to [18].

2.1 Generic Cryptographic Functions

The following definition has been adapted from [9, Def. 2.4.4].

Definition 1. *A collection of functions* $\{f_i : \{0,1\}^n \to \{0,1\}^n\}_{i \in I}$ *is called a* collection of trapdoor permutations (TDP) *if there exist four efficient algorithms* Ind, Dom, $Func$, $Func^{-1}$, *such that the following three conditions hold:*

1. Easy to sample and compute: *The output distribution of algorithm* Ind *on input* 1^N *is a random variable assigned values in the set* $(I \times T) \cap (\{0,1\}^N \times \{0,1\}^N)$*;* I *is called a set of* indices *and* T *a set of* trapdoors. *The output distribution of algorithm* Dom *on input* $i \in I$ *is a random variable assigned values in* $\{0,1\}^n$. *On input* $i \in I$ *and* $x \in \{0,1\}^n$, *algorithm* $Func$ *always outputs* $f_i(x)$.
2. Hard to invert: *There exists no efficient algorithm which on input* $(I_N, f_{I_N}(X_N))$ *returns* $f_{I_N}^{-1}(f_{I_N}(X_N))$, *except with probability negligible in* N, *where* I_N *is a random variable describing the distribution of the first element in the output of* Ind *on input* 1^N, *and* X_N *is a random variable describing the output of algorithm* Dom *on input (random variable)* I_N.
3. Easy to invert with trapdoor: *There exists a deterministic efficient algorithm* $Func^{-1}$, *such that for every* (i, t) *in the range of* Ind *and for every* $x \in \{0,1\}^n$, *it holds that* $Func^{-1}(t, f_i(x)) = x$.

A collection of *oneway permutations (OWP)* is defined analogously to the above, except that no trapdoor is specified (i.e. the algorithm Ind outputs only I) and no easy inversion is required (i.e. the third property is absent).

A collection of *trapdoor functions (TDF)* is defined analogously to Def. 1, except that the set of functions is $\{f_i : \{0,1\}^n \to \{0,1\}^*\}_{i \in I}$. In other words, the range may not coincide with the domain.

The following encryption function is a TDF candidate.

2.2 McEliece PKC

Technically speaking, we assume more than just a secure McEliece encryption. Instead, we take two assumptions which together imply its security [28] (while the opposite implication is not known to hold).

For the sake of completeness, let us briefly describe this PKC [19]. Define the space of public keys $PK \stackrel{def}{=} SGP$, where $S \in_R \{0,1\}^{k\times k}$ non-singular; $G \in \{0,1\}^{k\times n}$ a generating matrix of an irreducible Goppa code correcting t errors, $k \geq n - t \cdot \log_2 n$; $P \in \{0,1\}^{n\times n}$ random permutation matrix. The secret key is the decoding algorithm of the code G (loosely speaking, the knowledge of (S, G, P) is enough). Note that each $pk \in PK$ represents a linear (n, k) code.

Encryption of a message $m \in \{0,1\}^k$ proceeds by choosing a random weight-t vector $e \in \{0,1\}^n$ and computing the ciphertext $s = m(SGP) + e$. Decryption proceeds by first applying P^{-1} to s and then decoding according to G.

The problems underlying the following two assumptions are discussed in details in [28, Sec. 6], so we limit our presentation to a brief sketch. Let us denote by *Goppa-IND* the problem of distinguishing a randomly sampled McEliece public key from a random linear code (with the same parameters). The security parameter is n, the code length.

Assumption 1. *Goppa-IND is hard on the average.*

Goppa-IND is not known to reduce to any hard problem.

Let *Goppa Bounded Decoding (GBD)* be the problem of syndrome decoding with the following promise: the number of errors is guaranteed to be up to t, as in the definition of the Goppa code.

Assumption 2. *GBD is hard on the average.*

In other words, without knowing a structure of the code (as the previous assumption suggests), it is hard to decode errors in the corresponding codeword (and hence, to invert the McEliece encryption). The underlying problem is not known to be NP-complete, but it is related to the Bounded Distance Decoding problem conjectured to be NP-hard, and, in turn, the later is connected to NP-complete Syndrome Decoding problem.

Remark 1. *Assumption 1 implies that one can perform the McEliece encryption on $[\mathbf{I}^k|M]$, $M \in_R \mathcal{C}$ without noticing (except with negligible probability) that the encryption key does not belong to PK. Indeed, otherwise a (trivial) distinguisher for the McEliece public keys could use the encryption algorithm.*

Note that the aforementioned cryptographic function is hard-to-invert by Assumption 2.

For our proofs, we need an additional ad-hoc assumption. We call a subset of (n, k) linear codes (denoted by $\mathcal{C}_t$) *trapdoor Goppa-Bound decodable (trapdoor-GBD)*, if the following holds on the average for all the codes in this subset:

- GBD problem is hard given a particular representation of the code.
- GBD problem is easy given the code's representation and some auxiliary input of size polynomial in the security parameter n.

In the above definition, we intend to cover all linear codes with the properties similar to that of the irreducible Goppa codes.[3]

Assumption 3. *$|\mathcal{C}_t|/|\mathcal{C}|$ is negligible in the security parameter n.*

Let us now provide two evidences in support of this assumption. First, it is easy to check given the bound on the number of the McEliece public keys [28, Sec. 2.2.2] that the following claim is true.

Claim 1. *$|PK|/|\mathcal{C}|$ is negligible in n.*

[3]Indeed, it is hard to decode its permuted version (and hence to invert the McEliece encryption) just knowing the public key. While the secret key (the trapdoor) allows such the decoding.

Moreover, to our best knowledge, any family of good codes (known so far) represents a negligibly small (in n) fraction of $\mathcal{C}$.

Secondly, any substantially large (i.e. of size at least superpolynomial in n) family in the subset $\mathcal{C}_t$ is a potential candidate to be used in a coding based PKC. Search for such the candidates has not been very successful so far [7, Sec. 1.1].

2.3 Interactive hashing

IH is a two-party cryptographic primitive. It takes as input a string $w \in \{0,1\}^n$ from a sender S, and produces as output two n-bit strings, one of which is w and the other $w' \neq w$ (let us call w and w' the *first* and the *second* output, respectively). The output strings are available to both S and a receiver R, such that, loosely speaking, a dishonest sender $\widetilde{\mathsf{S}}$ cannot control both of them, while a dishonest receiver $\widetilde{\mathsf{R}}$ cannot tell which one was the input.

The (dis)honest sender/receiver in a two-party protocol will be denoted henceforth in this paper as above.

For the following definitions, we borrow notation from [16, Sec. 3.1]. Let $\mathcal{H} : \{0,1\}^n \to \{0,1\}^{n-1}$ be a family of two-to-one Boolean hash functions and $g : \{0,1\}^n \to \{0,1\}^n$ be an OWP. Both players receive the security parameter 1^n as an input and S gets as a private input $y \in \{0,1\}^n$. At the end, S locally outputs y and, in addition, both S and R output $(h, z) \in \mathcal{H} \times \{0,1\}^{n-1}$.

Definition 2 (Adapted from [27], Def. 2.1)**.** Information-theoretic interactive hashing *satisfies the following three properties.*

1. Correctness: *For all n, all $y \in \{0,1\}^n$, and every pair $(y,(h,z))$ that may be output by $(\mathsf{S}(1^n, y), \mathsf{R}(1^n))$, it is the case that $h(y) = z$.*
2. Hiding: *There exists a polynomial-time simulator Sim such that for every $y \in \{0,1\}^n$ and $h \in \mathcal{H}$ the distributions $View_{\mathsf{R}}(\mathsf{S}(y), \mathsf{R})(1^n)$ and $Sim(1^n, h, h(y))$ are identical.*
3. Binding: *Let $T \subset \{0,1\}^n$. No $\widetilde{\mathsf{S}}$ succeeds in the following game with probability more than $O(|T|/2^n)$. On security parameter 1^n, $\widetilde{\mathsf{S}}$ interacts with R and R outputs pairs (y_0, y_1) such that $y_0, y_1 \in T$ and $h(y_0) = h(y_1) = z$.*

The following definition (focusing on a particular scenario of [21]) for the computational flavor of IH is adapted from [16, Sec. 3.1].

Definition 3. Computational interactive hashing *satisfies the following three properties.*

1. Correctness: *Identical to that of Def. 2.*
2. Hiding: *Identical to that of Def. 2.*
3. Binding: *No PPT $\widetilde{\mathsf{S}}$ succeeds in the following game with more than negligible probability. On security parameter 1^n, $\widetilde{\mathsf{S}}$ interacts with R and R outputs pairs $(x_0, y_0), (x_1, y_1)$ such that $y_0 = g(x_0)$, $y_1 = g(x_1)$ and $h(y_0) = h(y_1) = z$.*

In our work, we abstract the implementation details as much as possible and use IH as a blackbox. Any IH protocol which satisfies one of the above definitions fits one of our constructions.

Concrete protocols realizing C-IH and IT-IH can be found, e.g., in [21, Section 3.1] (appears as a part of the committing stage) and [27, Protocol 2.1], respectively. These protocols are very similar, the main difference between them is in construction of receiver's queries (refer to Appendix A for description and some details). The main disadvantage of these protocols is their round and communication complexity: for n-bit input, they require $n-1$ rounds and $O(n^2)$ bits of communication.[4]

[4]There exists a constant round (information-theoretic) IH protocol [5, Sec. 5] with roughly the same communication cost. However, its security requirements are somewhat stricter compared to ours, so that we cannot use it in our schemes. On the bright side, this result shows us that improvements are possible in principle.

2.4 Oblivious transfer

We focus on 1-out-of-2 bit oblivious transfer [8]: The honest sender S transmits two input bits b_0, b_1 such that one of them b_c is obtained by the honest receiver R according to his input - the choice bit c. The dishonest sender $\widetilde{\mathsf{S}}$ cannot learn c, while the dishonest receiver $\widetilde{\mathsf{R}}$ cannot learn both b_0 and b_1.

In our schemes, c is chosen independently of an honest receiver, in the course of the protocol execution. Thus, in order to achieve 1-2 OT, the players must take the following two-step procedure. First, they execute one instance of our protocol with uniformly random inputs, which results in "pre-computing" oblivious transfer. Second, they transform this "pre-computed" OT into the 1-2 OT (where they actually choose their inputs) using the (very efficient) reduction of [1, Sec. 3.2].[5] Therefore, in the security argument for our protocols, we will disregard attacks, where $\widetilde{\mathsf{R}}$ tries to bias the choice of c. Indeed, the honest receiver will achieve the same using the aforementioned reduction.

Our protocols involve the sender transmitting two encryptions of his corresponding inputs. One standard sender's attack is to render one of the encryptions invalid, hoping that R encounters a decrypting error, complains and hereby reveals his choice. Clearly, this attack is fruitless in our scenario, where the choice bit is random.

Definition 4 (Adapted from [6], Def. 1)**.** *A protocol* $[\mathsf{S}, \mathsf{R}](b_0, b_1)$ *is said to* securely implement randomized oblivious transfer*, if at the end of its execution by the sender* S *and the receiver* R *which are represented by PPT algorithms having as their input a security parameter* n*, such that the following properties hold:*

- Correctness: *when the players honestly follow the protocol,* R *outputs* (c, b_c) *for* $c \in_R \{0,1\}$ *while* S *has no output.*
- Sender-security: *For every PPT adversary* $\widetilde{\mathsf{R}}$*, every input* z*, a (sufficiently long) random tape* R_{S}*, there exists a choice bit* c *such that for* $b_c \in \{0,1\}$ *the distribution (taken over* S*'s randomness) of runs of* $\widetilde{\mathsf{R}}(z)$ *using randomness* R_{S} *with* S *having input* b_c *and* $b_{1-c} = 0$ *is computationally indistinguishable from the distribution of runs with* S *having input* b_c *and* $b_{1-c} = 1$*.*
- Receiver-security: *For any PPT adversary* $\widetilde{\mathsf{S}}$*, any security parameter* n *and any input* z *of size polynomial in* n*, the view that* $\widetilde{\mathsf{S}}(z)$ *obtains when* $c = 0$ *is computationally indistinguishable from that of when* $c = 1$*, denoted:* $\{View_{\widetilde{\mathsf{S}}}(\widetilde{\mathsf{S}}(z), \mathsf{R})\}_z \stackrel{c}{=} \{View_{\widetilde{\mathsf{S}}}(\widetilde{\mathsf{S}}(z), \mathsf{R})\}_z$.

2.5 Hardcore Bit Encryption

Let x and r be bit vectors of appropriate length and denote the scalar product by "$\cdot$". It follows from [10] that for any TDF f, the product $x \cdot r$ is a *hardcore bit*, i.e. it cannot be guessed substantially better than at random, given $(f, f(x), r)$.

Then, encrypting a plaintext bit b given a trapdoor function f amounts to generating (x, r) at random and computing the ciphertext as the triple $(b + (x \cdot r), f(x), r)$, where "+" is an exclusive-or. Decryption is performed in a straight forward way, inverting f using its trapdoor. Clearly, without the trapdoor, b is as hard to guess as the hardcore bit of f.

Note that TDP, as a particular case of TDF, fits this construction as well.

3 Efficiently Reducing OT to Trapdoor Permutations

Let us assume a secure (according to Def. 3) implementation of C-IH (in this section, we refer to it as "IH" for short) to be available as a blackbox. Let $\{f_i : \{0,1\}^n \to \{0,1\}^n\}_{i \in I}$ be a TDP collection according to Def. 1. Let us define by $f_i^h(x, b)$ a hardcore bit encryption

[5] Due to space limitations, we omit further details. The reader may consult [6, Sec. 4.2] for description.

(as described in the previous section) of a bit b using a permutation f_i as trapdoor function and some $x \in \{0,1\}^n$.

S has inputs $b_0, b_1 \in_R \{0,1\}$, R has no input.

Protocol 1.

1. S generates $i \in_R I$ and the corresponding trapdoor t, then sends i to R.
2. R generates $x \in_R \{0,1\}^n$, computes $y = f_i(x)$, then inputs y to IH, such that both S and R obtain (y_0, y_1) as output.
   The outputs are assigned to y_j according to lexicographic order.
3. For $j = 0, 1$: S uses t to compute $x_j = f_i^{-1}(y_j)$ and sends $f_i^h(x_j, b_j)$ to R.
   R computes $c = \{j | x_j = x\}$, decrypts b_c and outputs (c, b_c).

Proposition 1. *Protocol 1 is a secure implementation of 1-2 OT according to Definition 4 with information-theoretic receiver-security, assuming that trapdoor permutations exist.*

sketch. **Correctness.** When both players are honest, one of the outputs of IH is indeed R's input y by the IH-correctness property. Then in Step 3, R correctly identifies y_c and successfully decrypts the corresponding S's input b_c.

Sender-security. Due to the use of hardcore bit encryption, $\widetilde{\mathsf{R}}$ must invert f_i on both y_0 and y_1, in order to learn both b_0 and b_1. However, the binding property of Def. 3 denies it under assumption that TDP exist.

Receiver-security. Computationally unbounded $\widetilde{\mathsf{S}}$ must tell y among (y_0, y_1) to successfully learn c. However, f_i is a permutation and hence, if x is uniformly distributed in D_i, then so is $f_i(x) = y$. Now, the hiding property of Definition 3 denies $\widetilde{\mathsf{S}}$'s successful attack. □ □

3.1 Extension to Trapdoor Functions

A natural question is whether in the above protocol, one can substitute TDP with trapdoor *functions*. An immediate problem here is that their range does not coincide with their domain. Hence, encryption of a uniformly sampled domain element does not produce a uniform range element. Moreover, when the range is larger than the domain (which is typical for TDF candidates), IH may produce a (recognizable) invalid encryption on the output, hence revealing R's choice.

Next, we very informally describe an approach which may allow us to build OT using TDF with some special properties. Let $\{f_i : \{0,1\}^n \to \{0,1\}^*\}_{i \in I}$ be a collection of TDF. Let R choose $i \in_R I$ and input it into IH such that both S and R obtain (i_0, i_1). Now, S encrypts his inputs b_0 and b_1 using hardcore bit encryption with f_{i_0} and f_{i_1}, respectively (the corresponding x will be generated by S at random). Note that as long as I can be succinctly represented by binary strings (i.e. for some integer l, there exists a one-to-one correspondence between I and $\{0,1\}^l$), information-theoretic receiver-security would hold similarly to Protocol 1. Unfortunately, the sender-security proof of Protocol 1 does not apply directly, and we could not construct a convincing argument to replace it.[6]

Taking a way around this problem, we introduce the following OT protocol which is based on the assumptions underlying the McEliece cryptosystem (a TDF candidate) and information-theoretic version of IH.

4 OT from the McEliece Assumptions

In this section, we restrict our consideration to the range of (n, k), which is relevant to the secure McEliece PKC with parameters (n, t) as described in Sec. 2.2.

[6]On the other hand, it is quite intuitive that inversion of some function from the TDF collection must be hard, if adversary has little control over the choice of this function. "Little control" is in the sense of interactively hashing the function's index.

Let us assume a secure (according to Def. 2) implementation of IT-IH (in this section, we call it "IH" for short) to be available as a blackbox. When a binary matrix is input into IH, it is represented as a bit string by concatenating the rows.

Denote the standard part of a generating $k \times n$ matrix M by $St(M)$. Let $Enc_{pk}(b)$ be a hardcore bit encryption (according to Sec. 2.5) of a bit b where trapdoor function is the McEliece encryption having a public key $pk \in PK$. The string x is generated uniformly at random by the encrypting player, its mentioning is omitted for the sake of notation simplicity.

The sender S has inputs $b_0, b_1 \in_R \{0,1\}$, the receiver R has no input.

Protocol 2.

1. a) R generates a random $pk \in PK$, computes $St(pk)$ and inputs the latter into IH, then both S and R obtain outputs which are assigned to (w_0, w_1) according to lexicographic order.
   b) For $i = 0, 1$: Both players parse w_i as an element of $\{0,1\}^{k \times n-k}$ and compute $K_i = [\mathbf{I}^k | w_i]$.
2. For $i = 0, 1$: S sends $Enc_{K_i}(b_i)$ to R.
   R computes $c = \{i | K_i = pk\}$, decrypts b_c and outputs (c, b_c).

Remark 2. *We believe, but do not prove formally, that the Niederreiter PKC [22] can be used in the above protocol in a similar manner.*

Proposition 2. *Protocol 2 is a secure implementation of 1-2 OT according to Definition 4 under Assumptions 1-3.*

sketch. **Correctness.** When both players are honest, one of the outputs of IH is indeed (an equivalent representation of) pk by the correctness property of Def. 2. Then in Step 2, R (knowing both K_0 and K_1) correctly identifies K_c, computes c and successfully decrypts b_c.

For the following discussion, it is worth noting that the set of linear code representations (by standard parts) $\mathcal{C} = \{0,1\}^{k \times n-k}$, hence the input to IH is of length $k(n-k)$ bits. Therefore, the parameter n in Def. 2 must be replaced with $k(n-k)$.

Sender-security. Due to the use of hardcore bit encryption, $\widetilde{\mathsf{R}}$ succeeds in learning something about both b_0 and b_1 (in the computational sense), only if it can invert encryption on both K_0 and K_1. Hence, it is sufficient to show that $\widetilde{\mathsf{R}}$ is unable to steer both K_0 and K_1 into a "bad" subset (denote it by $\mathcal{C}_b$) of codes which are efficiently Goppa-bounded decodable. By the binding property of Def. 2, $\widetilde{\mathsf{R}}$ has negligible probability of success as long as $|\mathcal{C}_b|/|\mathcal{C}|$ is negligible in n.

Let us observe that $\mathcal{C}_b$ consists of two non-intersecting subsets: 1) All codes which are efficiently GBD-decodable given only their description (denote them by $\mathcal{C}_e$) and 2) Trapdoor-GBD codes $\mathcal{C}_t$ (as defined in Sec. 2.2). We argue next that either subset's size is a negligible (in n) fraction of $|\mathcal{C}|$.

Lemma 1. *Under Assumption 2, $|\mathcal{C}_e|/|\mathcal{C}|$ is negligible in n.*

Proof. The opposite suggests that with non-negligible probability, for H – as in Assumption 2 – which is chosen uniformly at random, its corresponding generating matrix $H^{\perp} \in \mathcal{C}'$. In other words, such the code is efficiently decodable, a contradiction. □ □

As for $\mathcal{C}_t$, it's fraction is negligible by Assumption 3.

Remark 3. *Here, it is in order to clarify why the constant-round IH of [5] cannot be directly used in our protocol. Their security proof requires a priori knowledge of the upper bound on the "bad" subset size. Such an upper bound seems to be hard to obtain for linear codes.*

Receiver-security. The only way for $\widetilde{\mathsf{S}}$ to distinguish his views for different value of c (with non-negligible bias) is to tell the McEliece public key among (K_0, K_1). In the following, we show this attack to be impossible.

First of all, note that a passive $\widetilde{\mathsf{S}}$ cannot succeed in distinguishing K_0 and K_1. This is because the McEliece public key K_i (for $i \in \{0,1\}$) is indistinguishable by Assumption 1, while K_{1-c} will be uniform in $\mathcal{C}\backslash$ by the correctness property of Def. 2.

Thus, the IH protocol execution must complete with $\widetilde{\mathsf{S}}$ steering the second input into some efficiently recognizable subset $ER \subset \mathcal{C}$. Note that ER substantially (for more than a negligible fraction) intersecting with PK contradicts to Assumption 1.

Let us show, for two possible cases, that existence of such EC contradicts to our assumptions.

Case 1: $|ER|/|\mathcal{C}|$ is non-negligible in n. It contradicts to Assumption 1, since a uniformly chosen linear code has a non-negligible probability to hit $EC \setminus PK$. At the same, it has a negligible probability to hit the public key (see Claim 1). This implies distinguishing of public keys from random codes.

Case 2: $|ER|/|\mathcal{C}|$ is negligible in n. This implies that there must necessarily exist an efficient algorithm Alg which, by playing for a receiver in IH, can steer the second output to EC (with non-negligible probability), whenever the input is in PK. Let us construct a distinguisher D which given two matrices $(M_i \in_R PK, M_{i-1} \in_R \mathcal{C})$ for some $i \in_R \{0,1\}$ will output i (with non-negligible probability). D runs two copies of the IH protocol, having Alg to play for a sender in both and simulating an honest receiver who, respectively, inputs M_j in the j-th copy, for some $j \in \{0,1\}$. We emphasize that D keeps Alg completely ignorant of what is input into either IH instance. D outputs i for the i-th copy of IH, such that on input M_i, at least one of the IH outputs is in EC. Let us briefly argue correctness of D. For $M_i \in_R PK$, Alg is likely to succeeds in steering the second output into EC. On the other hand, for $M_i \in_R \mathcal{C}$, Alg will most likely fail due to the hiding property of Def. 2. We conclude that D correctly distinguishes the McEliece public key, hence contradicting to Assumption 1. □ □

Remark 4. *In fact, we cannot exclude that some proof similar to that of [21] will work for our protocol as well. However, in this work, we are mostly after the proof of concept. Thus, we chose to take a shortcut by admitting Assumption 3, rather than constructing the whole proof in the computational IH scenario.*

5 Suggestions for Future Work

String Oblivious Transfer. Note that up to $\log k$ hardcore bits (where k is the range size of the underlying trapdoor function) are available in the hardcore bit encryption [10]. Hence, it is easy to generalize our constructions to oblivious transfer of (short) strings instead of bits.

Interactive Hashing. It is interesting to look for new natural applications of IH in cryptography. Another important question is efficiency of IH protocols. Although it was shown in [14] that based on OWP in blackbox manner, IH must have $\Omega(n/\log n)$ rounds, this question remains open for specific computational assumptions.

Relaxing Security Assumptions for TDP-based Protocol. Can one (efficiently) employ standard TDF there? If not, then what are minimal assumptions for TDF to be applicable?

Generalizing Coding Based Protocol. It is interesting to generalize Protocol 2 for the case of any PKC with the public key indistinguishable from random (e.g., lattice based PKC's).

References

[1] D. Beaver. Precomputing oblivious transfer. In D. Coppersmith, editor, *CRYPTO*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 1995.

[2] C. Cachin, C. Crépeau, and J. Marcil. Oblivious transfer with a memory-bounded receiver. In *FOCS*, pages 493–502, 1998.

[3] R. Canetti, editor. *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*. Springer, 2008.

[4] C. Crépeau. Equivalence between two flavours of oblivious transfers. In C. Pomerance, editor, *CRYPTO*, volume 293 of *Lecture Notes in Computer Science*, pages 350–354. Springer, 1987.

[5] Y. Z. Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-round oblivious transfer in the bounded storage model. *J. Cryptology*, 20(2):165–202, 2007. Conference version appears at TCC '04.

[6] R. Dowsley, J. van de Graaf, J. Müller-Quade, and A. C. A. Nascimento. Oblivious transfer based on the mceliece assumptions. In R. Safavi-Naini, editor, *ICITS*, volume 5155 of *Lecture Notes in Computer Science*, pages 107–117. Springer, 2008.

[7] D. Engelbert, R. Overbeck, and A. Schmidt. A summary of McEliece-type cryptosystems and their security. *Journal of Mathematical Cryptology*, 1(2):151–199, 2007.

[8] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.

[9] O. Goldreich. *Foundations of Cryptography - Volume 2 (Basic Applications)*. Cambridge University Press, 2004.

[10] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

[11] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229. ACM, 1987.

[12] I. Haitner. Implementing oblivious transfer using collection of dense trapdoor permutations. In M. Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 394–409. Springer, 2004.

[13] I. Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In Canetti [3], pages 412–426.

[14] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *FOCS*, pages 669–679. IEEE Computer Society, 2007.

[15] I. Haitner, J. J. Hoch, and G. Segev. A linear lower bound on the communication complexity of single-server private information retrieval. In Canetti [3], pages 445–464.

[16] I. Haitner and O. Reingold. A new interactive hashing theorem. In *IEEE Conference on Computational Complexity*, pages 319–332, 2007.

[17] Y. T. Kalai. Smooth projective hashing and two-message oblivious transfer. In R. Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 78–95. Springer, 2005.

[18] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library. Elsevier, 12 edition, 2006.

[19] R. J. McEliece. A public key cryptosystem based on algebraic coding theory. *DSN progress report*, 42-44:114–116, 1978.

[20] J. Müller-Quade. Personal communication, December 2008.

[21] M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for p using any one-way permutation. *J. Cryptology*, 11(2):87–108, 1998.

[22] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Probl. Control and Inform. Theory*, 15:19–34, 1986.

[23] R. Ostrovsky, R. Venkatesan, and M. Yung. Fair games against an all-powerful adversary. In *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 155–169, 1993.

[24] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.

[25] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *STOC*, pages 187–196. ACM, 2008.

[26] M. O. Rabin. How to exchange secrets by oblivious transfer. Technical report, Aiken Computation Laboratory, Harvard University, 1981. Tech. Memo TR-81.

[27] G. Savvides. *Interactive Hashing and reductions between Oblivious Transfer variants.* PhD thesis, School of Computer Science, McGill University, Montreal, Canada, 2007.

[28] N. Sendrier. On the security of the McEliece public-key cryptosystem. In M. Blaum, P. G. Farrell, and H. C. A. van Tilborg, editors, *Information, Coding and Mathematics*, pages 141–163. Kluwer, 2002. Proceedings of Workshop honoring Prof. Bob McEliece on his 60th birthday.

[29] S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.

Appendix A Interactive Hashing Protocol

The following protocol was introduced in [23]. Let w be a n-bit string that a sender S wishes to transmit to a receiver R. All operations take place in the binary field $\mathbb{F}_2$.

Protocol 3. Interactive Hashing

1. R chooses a $(n-1) \times n$ matrix Q uniformly at random among all binary matrices of rank $n-1$. Let q_i the i-th query, consisting of the i-th row of Q.
2. For $1 \leq i \leq n-1$ do:
   (a) R sends query q_i to S.
   (b) S responds with $c_i = q_i \cdot w$.
3. Given Q and c (the vector of R's responses), both parties compute the two values of w consistent with the linear system $Q \cdot w = c$. These solutions are labeled w_0, w_1 according to lexicographic order.

A proof that the above protocol is IT-IH (i.e. satisfies Definition 2) can be found in [27, Sec. 2.3]. A slight modification of Protocol 3 is shown to satisfy Definition 3 (i.e. to be C-IH) in [21], with a tighter security proof presented in [16] (along with several generalizations). The modification (compared to Protocol 3) consists of choosing the matrix Q in a canonical way.