Oblivious Transfer Based on the McEliece Assumptions with Unconditional Security for the Sender

Bernardo M. David[1], **Anderson C. A. Nascimento**[1], **Rodrigo B. Nogueira**[1]

[1]Department of Electrical Engineering, University of Brasilia.
Campus Universitario Darcy Ribeiro,Brasilia, CEP: 70910-900, Brazil

bernardo.david@redes.unb.br, andclay@ene.unb.br, rodrigo.b.nogueira@redes.unb.br

Abstract. *In this paper we propose the first code-based oblivious transfer protocol with perfect (unconditional) security for one of the parties. To obtain this result we show that the McEliece cryptosystem is rerandomizable, a property that might be of independent interest.*

1. Introduction

Oblivious Transfer [18][8] is an important cryptographic primitive which implies secure two-party computation [12][10] as well as multi-party computation [4]. In the present work, we focus on the one-out-of-two oblivious transfer (OT). This is a two-party primitive where a $sender$ (Alice) inputs two bits b_0, b_1 and a $receiver$ (Bob) inputs a bit c, referred to as the *choice bit*. Bob learns b_c but must not learn any information on b_{1-c} and b_c *simultaneously* while Alice learns nothing about Bob's choice (the value of c).

OT has been constructed based on various computational assumptions, such as enhanced trapdoor permutations [7], hardness of factoring [18], Diffie-Hellman [2], Quadratic or Higher-Order Residuosity and the Extended Riemman Hypothesis [11]. In all of these constructions the security for one of the players is unconditional, that is, the security for one of the players holds even when she is computationally unbounded. Recently, constructions of oblivious transfer based on coding theory hypotheses have been independently proposed in [6] and in [13]. In these cases, differently from previous results, the security for both players is computational, even against passive adversaries. Moreover, it was stated as an open problem, in [6] and in [13], to obtain code-based oblivious transfer protocols with unconditional security for one of the parties. We address this issue in this paper.

We implement 1-out-of-2 OT based on the two assumptions of the McEliece PKC [14]: the hardness of decoding random binary linear codes (which was proven to be NP-Complete [3] and equivalent to the Learning Parity with Noise (LPN) problem [19]) and the indistinguishability of the scrambled generating matrix of a binary Goppa code from a random one [15]. Particularly, we achieve unconditional security for Alice and computational security for Bob, thus solving the open problem stated in [6] and in [13]. To achieve our result we introduce a novel property of the McEliece public key cryptossystem (PKC) which, to the best of our knowledge, has not been presented in the literature: we show that the McEliece PKC is re-randomizable. An encryption scheme is said to be re-randomizable when it is possible to change its randomness without knowing its private key. We believe this property of the Mceliece PKC is of independent interest and might find other applications elsewhere.

In this work we consider only *static* adversaries, i.e., either Alice or Bob is corrupted before the protocol starts.

2. Preliminaries

In this section we define the notation used throughout this work and provide the formal security definitions for oblivious transfer and bit commitment. We also describe the McEliece public key cryptosystem and the security assumptions on which both this PKC and our protocol are based.

Hereupon, we will denote by $x \in_R D$ an uniformly random choice of element x over its domain D; by $\oplus$ a bit-wise exclusive OR of strings; and by $a \mid b$ the concatenation of string a with string b. All logarithms are to the base 2.

Two sequences $X_n, n \in \mathbb{N}$ and $Y_n, n \in \mathbb{N}$ of random variables are said to be *computationally indistinguishable*, denoted by $X \stackrel{c}{=} Y$, if for every non-uniform probabilistic polynomial-time distinguisher D there exists a negligible function $\epsilon(\cdot)$ such that for every $n \in \mathbb{N}$,

$$\mid Pr[D(X_n) = 1] - Pr[D(Y_n) = 1] \mid < \epsilon(n)$$

2.1. McEliece Public-Key Cryptosystem

The following definition has been taken from [17]. The McEliece cryptosystem [14] consists of a triplet of probabilistic algorithms $ME = (Gen_{ME}, Enc_{ME}, Dec_{ME})$ and $M = \{0,1\}^k$.

- Key generation algorithm: The `PPT` key generation algorithm Gen_{ME} works as follows:
  1. Generate a $k \times n$ generator matrix $\mathbf{G}$ of a binary Goppa code, where we assume that there is an efficient error-correction algorithm `Correct` which can always correct up to *w* errors.
  2. Generate a $k \times k$ random non-singular matrix $\mathbf{S}$.
  3. Generate a $n \times n$ random permutation matrix $\mathbf{P}$.
  4. Set $\mathbf{G} = \mathbf{SG'P}$, and outputs $pk = (\mathbf{G}, w)$ and $sk = (\mathbf{S}, \mathbf{G'}, \mathbf{P})$.
- The encryption algorithm: The `PPT` encryption algorithm Enc_{ME} takes a plaintext $m \in \{0,1\}^k$ and the public-key pk as input and outputs ciphertext $c = m\mathbf{G} \oplus e$, where $e \in \{0,1\}^n$ is a random vector of hamming weight w.
- The decryption algorithm: The polynomial-time algorithm Dec_{ME} works as follows:
  1. Compute $c\mathbf{P}^{-1} = ((m\mathbf{S})\mathbf{G'} \oplus e\mathbf{P}^{-1})$, where $\mathbf{P}^{-1}$ denotes the inverse matrix of $\mathbf{P}$.
  2. Compute $m\mathbf{S} = \texttt{Correct}(c\mathbf{P}^{-1})$.
  3. Output $m = (m\mathbf{S})\mathbf{S}^{-1}$.

2.2. Semantically Secure McEliece Public-Key Cryptosystem

In our application we use a variant of the original McEliece PKC that was proved semantically secure [17]. We are interested in encrypting single bit messages, instead of strings.

In the semantically secure version of McEliece PKC, we pad the message $m \in \{0,1\}$ with a random string $r \in_R \{0,1\}^{k-1}$, obtaining $(r \mid m)\mathbf{G} \oplus e$ as the desired ciphertext.

Additionally, instead of creating an error vector by choosing it randomly from the set of vectors with Hamming weight w, we generate e by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. Clearly, due to the law of large numbers, the resulting error vector should be within the error capabilities of the code.

2.3. Re-randomization of the McEliece Public-Key Cryptosystem

A public key cryptosystem is said to be re-randomizable if one is able to change the randomness of a given ciphertext without knowing its correspondent secret key. In the case of the McEliece PKC, one can obtain a form of weak re-randomization, which will be enough for our purposes. By weak, we mean that one will be able to change the randomness of a given cyphertext a constant number of times.

We call by *re-randomize* the process where, given a ciphertext $K \in \{0,1\}^n$, we compute the bitwise-XOR of K with $(r \mid 0)\mathbf{G} \oplus e$, where $r \in_R \{0,1\}^{k-1}$, i.e, $\hat{K} = K \oplus (r \mid 0)\mathbf{G} + e$, where e is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$.

Since the code in question is linear and the total number of errors added to the message is strictly less than w (with overwhelming probability due to the Chernoff Bound), the re-randomization preserves the value of $Dec_{ME}(\hat{K})$, *i.e.* $Dec_{ME}(\hat{K}) = Dec_{ME}(K)$, but turns $\hat{K}$ indistinguishable from a random vector of the same size due to the semantic security of the modified McEliece PKC.

2.4. Security Assumptions

In order to prove the security of our protocol, we use the McEliece PKC assumptions, which will be introduced and briefly discussed in this subsection. The first assumption is that there is no algorithm which can distinguish the scrambled generating matrix of a binary Goppa code (matrix G generated by the algorithm Gen_{ME} described before) from a random matrix of the same size.

Assumption 1. *There is no PPT algorithm which can distinguish the public-key matrix P of the McEliece cryptosystem from a random matrix of the same size with non-negligible probability.*

The hardness of decoding random linear codes is the second assumption on which the McEliece PKC and our protocol are based. This problem was proved to be NP-complete [3] and currently the best algorithm to solve this problem still runs in exponential time. This problem was proved to be equivalent to the Learning Parity with Noise (LPN) problem [19], a fact that is central to our scheme's security.

Assumption 2. *The Syndrome Decoding Problem is hard for every PPT algorithm.*

2.5. Security Definition of Oblivious Transfer

This definition was taken from [6] and slightly adapted to protocols with unconditional security for Alice. Let us denote by $View_{\tilde{A}}(\tilde{A}(z), B(c))$ and $View_{\tilde{B}}(A(b_0, b_1), \tilde{B}(z))$ the *views* of dishonest Alice and Bob, respectively, which represent their inputs z, results of all local computations, and messages exchanged.

Definition 1. *A protocol* $[A, B](b_0, b_1; c)$ *is said to securely implement oblivious transfer, if at the end of its execution by the sender Alice and the receiver Bob which are modeled as probabilistic polynomial time (PPT) Turing machines having as their input a security parameter N, the following properties hold:*

- *Completeness: when the players honestly follow the protocol, Bob outputs* b_c *while Alice has no output.*
- *Security for Alice: For every PPT adversary* $\tilde{B}$*, every input* z*, and a (sufficiently long) random tape* R_B *chosen at random, there exists a choice bit* c *such that for* $b_c \in \{0, 1\}$ *the distribution (taken over Alice's randomness) of runs of* $\tilde{B}(z)$ *using randomness* R_B *with Alice having input* b_c *and* $b_{\bar{c}} = 0$ *is indistinguishable from the distribution of runs with Alice having input* b_c *and* $b_{\bar{c}} = 1$*. Put differently:*

$$View_{\tilde{B}}(A(b_c, b_{\bar{c}=0}), \tilde{B}(z)) \mid_z \approx View_{\tilde{B}}(A(b_c, b_{\bar{c}=1}), \tilde{B}(z)) \mid_z$$

- *Security for Bob: For any PPT adversary* $\tilde{A}$*, any security parameter* N *and any input* z *of size polynomial in* N *, the view that* $\tilde{A}(z)$ *obtains when Bob inputs* $c = 0$ *is computationally indistinguishable from that of when Bob inputs* $c = 1$*, denoted:*

$$View_{\tilde{A}}(\tilde{A}(z), B(0)) \mid_z \stackrel{c}{=} View_{\tilde{A}}(\tilde{A}(z), B(1)) \mid_z$$

A protocol is said to be *secure in the semi-honest model* if the previous definition holds in the case Alice and Bob follow the protocol, being considered to be *secure in the malicious model* if the previous definition holds in the case one of the parties doesn't follow the protocol. An oblivious transfer protocol is unconditionally secure against a player if the given properties hold even when this player is computationally unbounded, in other words, the protocol is unconditionally secure for a player if the given properties hold even when the other player (an adversary) is computationally unbounded.

2.6. Security Definition of String Commitment

We will need string commitment schemes in our construction of a fully secure protocol. In a string commitment protocol there are two stages: *Commit* and *Open*. In the first stage, *Commit*, the sender (Alice) inputs a bit-string b and provides the receiver (Bob) with evidence about it. Bob doesn't learn b until the second stage, *Open*, when Alice reveals her commitment to Bob, who can detect with high probability that she opened a value different from b. Let $View_{\tilde{A}}(\tilde{A}(z), B(c))$ and $View_{\tilde{B}}(A(b_0, b_1), \tilde{B}(z))$ denote the the *views* of dishonest Alice and Bob, respectively, which represent their inputs z, results of all local computations, and messages exchanged. The following definition is based on [16] and [6]:

Definition 2. *A protocol* $[A, B](b_0, b_1; c)$ *is said to securely implement string commitment, if at the end of its execution by the sender Alice and the receiver Bob which are modeled as probabilistic polynomial time (PPT) Turing machines having as their input a security parameter N, the following properties hold:*

- *Completeness: when the players honestly follow the protocol, Bob accepts* b*.*
- *Hiding: For any PPT adversary* $\tilde{B}$*, any security parameter N and any input* z *of size polynomial in N, any* $k \in \mathbb{N}$*, after the Commit stage, but before the Open*

stage, the view of $\tilde{B}(z)$ when Alice inputs $b \in \{0,1\}^k$ is computationally indistinguishable from the view where Alice inputs any other $b' \in \{0,1\}^k$, $b' \neq b$:

$$View_{\tilde{B}}(A(b), \tilde{B}(z)) \mid_z \stackrel{c}{=} View_{\tilde{B}}(A(b'), \tilde{B}(z)) \mid_z$$

- *Binding: For any PPT adversary $\tilde{A}$, any security parameter N and any input z of size polynomial in N, any $k \in \mathbb{N}$, there exists $b \in \{0,1\}^k$ which can be computed by Alice after the Commit stage, such that the probability that $\tilde{A}(b')$, $b' \neq b$ is accepted by Bob in the Open stage is negligible in N.*

A string commitment protocol is unconditionally secure against a player if the properties in Definition 2 hold even when this player is not computationally bounded.

A bit commitment scheme can be constructed using a pseudorandom generator [16]. A pseudorandom number generator based on syndrome decoding was proposed in [9].

3. A Protocol Secure in the Semi-Honest Model

First we assume that Alice and Bob are honest-but-curious, that is they follow the protocol but try to obtain as much information as possible from their views. Bob sends to Alice a McEliece public key $\mathbf{G}$ and two pairs (K_0^0, K_1^0) and (K_0^1, K_1^1) of messages $(0, 1-c)$ and $(0, c)$ encrypted with a semantically secure McEliece cryptosystem [17] and $\mathbf{G}$, e.g, $K_0^0 = (r \mid 0)\mathbf{G} \oplus e$, where $r \in_R \{0,1\}^{k-1}$, e is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. Alice then encrypts b_0 and b_1 by re-randomizing $K_{b_0}^0$ and $K_{b_1}^1$, respectively, and sends the encryptions to Bob.

The protocol is complete because Bob can always decrypt b_c. The protocol is unconditionally secure for Alice because $Dec_{ME}(K_0^{1-c}) = Dec_{ME}(K_1^{1-c})$, thus Bob is unable to distinguish between $b_{1-c} = 0$ and $b_{1-c} = 1$. It is also computationally secure for Bob, because Alice cannot distinguish (K_0^0, K_1^0) from (K_0^1, K_1^1).

In this protocol Alice inputs two bits b_0 and b_1 while Bob inputs a bit c in order to receive b_c.

Protocol 1.

1. Bob generates $\mathbf{S}, \mathbf{G}', \mathbf{P}$, (K_0^0, K_1^0),(K_0^1, K_1^1) and sends $(\mathbf{G} = \mathbf{SG'P}, w)$, (K_0^0, K_1^0), (K_0^1, K_1^1) to Alice, where:
   - (S, G', P) is a McEliece secret key generated following the procedures of the McEliece Gen_{ME} algorithm and $(\mathbf{G} = \mathbf{SG'P}, w)$ is its corresponding public key.
   - (K_0^0, K_1^0) is a pair of encrypted messages such that $K_0^0 = (r_0 \mid 0)\mathbf{G} \oplus e_0$ and $K_1^0 = (r_1 \mid 1-c)\mathbf{G} \oplus e_1$, where $r_i \in_R \{0,1\}^{k-1}$, e_i is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. for $i = 0, 1$.
   - (K_0^1, K_1^1) is a pair of encrypted messages such that $K_0^1 = (r_0 \mid 0)\mathbf{G} \oplus e_0$ and $K_1^1 = (r_1 \mid c)\mathbf{G} \oplus e_1$, where $r_i \in_R \{0,1\}^{k-1}$, e_i is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. for $i = 0, 1$. for $i = 0, 1$

2. Alice computes for $i \in \{0,1\}$: $\hat{b_i} = K^i_{b_i} \oplus (r_i \mid 0)\mathbf{G}$, where $r_i \in_R \{0,1\}^{k-1}$ and sends $(\hat{b_0}, \hat{b_1})$ to Bob.
3. Bob decrypts $\hat{b_c}$ by executing the steps of the DEC_{ME} algorithm and extracts the last bit of the plain-text message (which is equal to b_c).

Theorem 1. *Protocol 1 is complete, unconditionally secure for Alice and computationally secure for Bob against passive attacks according to Definition 1 under Assumptions 1 and 2.*

Proof. Assuming that both players follow the protocol under passive attacks we prove the properties listed in Definition 1.

Completeness: Observing that Bob can always decrypt $\hat{b_c}$ and that it allows him to compute b_c it follows that the protocol is complete. Since binary Goppa codes are linear it follows that $\hat{b_c} = K^c_{b_c} \oplus (r_1 \mid 0)\mathbf{G}$, where $r_1 \in_R \{0,1\}^{k-1}$, is a valid encryption of the message $(r \oplus r_1 \mid b_c)$.

Unconditional Security for Alice: Let $\tilde{B}$ be any computationally unbounded cheating receiver. Let c be the bit such that $K^{1-c}_i = (r_i \mid 0)\mathbf{G} \oplus e_0$ for $i = 0, 1$. Note that $Dec_{ME}(\hat{b}_{1-c}) = (r \mid 0)$ for either $b_{1-c} = 0$ or $b_{1-c} = 1$, thus proving that the distribution (taken over Alice's randomness) of runs of $\tilde{B}(z)$ using randomness R with Alice having input b_c and $b_{\bar{c}} = 0$ is indistinguishable from the distribution of runs with Alice having input b_c and $b_{\bar{c}} = 1$.

Computational Security for Bob: Let $\tilde{A}$ be any PPT cheating sender. Since the randomized McEliece cryptosystem was proven to be IND-CPA secure under Assumptions 1 and 2 [17], it follows that $\tilde{A}$ cannot distinguish between K^i_j for $i, j \in \{0,1\}$. Furthermore $\tilde{A}$ cannot decrypt K^i_j and compute the message m, where $(r \mid m) = Dec_{ME}(K^i_j)$ for $i, j \in \{0,1\}$. Hence, the protocol *views* where Bob inputs $c = 1$ or $c = 0$ are computationally indistinguishable for Alice. □

While Protocol 1 is complete and secure in the Semi-Honest model it is still possible to implement attacks against it in the Malicious Model (when the parties deviate arbitrarily from the protocol specifications). A malicious Bob could send K^0_1 and K^1_1 to Alice such that $K^i_1 = (r_i \mid 1)\mathbf{G} \oplus e_0$. It is clear that this attack would enable Bob to learn both b_0 and b_1, breaking security for Alice.

4. A protocol for OT secure in the Malicious Model

To obtain a fully secure protocol, we need to somehow prevent Bob from sending Alice malformed keys in the previous protocol. Fortunately, the same problem has been treated previously in the literature [6]. We follow Dowsley *et al.* [6] with slight modifications in order to construct a protocol secure in the malicious model based on the passively secure Protocol 1.

1. Construct a protocol for randomized oblivious transfer in which Bob is forced to choose K^0_1 and K^1_1 such that $K^{1-c}_1 = (r \mid 0)\mathbf{G} \oplus e_0$, where $r \in_R \{0,1\}^{k-1}$, e is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. Otherwise he will be detected with probability at least $\frac{1}{2}$.

2. Convert the random oblivious transfer into oblivious transfer while retaining the same level of security.
3. Reduce the probability that a malicious Bob learns both b_0 and b_1.

4.1. Random OT with high probability of Bob cheating

We implement a protocol that outputs two random bits a_0,a_1 and outputs two random bits d,a_d to Bob. In this protocol, Alice can detect with probability at least $\frac{1}{2} - \epsilon$ that a malicious Bob has chosen K_1^0 and K_1^1 such that $K_1^i = (r_i \mid 1)\mathbf{G} \oplus e_0$, where $r_i \in_R \{0,1\}^{k-1}$, e_i is an error vector formed by choosing each of its bits according to the Bernoulli distribution $\mathcal{B}_\theta$ with parameter $\theta, \theta(1-\theta) < \frac{w}{2n}$. **Protocol 2**

1. Bob chooses $c_0, c_1 \in_R \{0,1\}$ and generates two McEliece secret keys $Sk_0 = (S_0, G_0', P_0)$ and $Sk_1 = (S_1, G_1', P_1)$. He then commits to Sk_0, c_0 and Sk_1, c_1 and stores $\mathbf{G_0} = S_0 G_0' P_0$ and $\mathbf{G_1} = S_1 G_1' P_1$.
2. Bob computes (K_0^0, K_0^0),(K_0^1, K_1^1) and $(\hat{K}_0^0, \hat{K}_1^0)$,$(\hat{K}_0^1, \hat{K}_1^1)$ where:
   - $K_0^0 = K_0^1 = K_1^{1-c_0} = (r \mid 0)\mathbf{G_0} \oplus e$,$K_1^{c_0} = (r \mid 1)\mathbf{G_0} \oplus e$.
   - $\hat{K}_0^0 = \hat{K}_0^1 = \hat{K}_1^{1-c_1} = (r \mid 0)\mathbf{G_1} \oplus e$,$\hat{K}_1^{c_1} = (r \mid 1)\mathbf{G_1} \oplus e$.

   Bob sends (K_0^0, K_0^0),(K_0^1, K_1^1),$(\hat{K}_0^0, \hat{K}_1^0)$, $(\hat{K}_0^1, \hat{K}_1^1)$,$G_0$,$G_1$,$w$ to Alice.
3. Alice chooses a challenge $j \in_R \{0,1\}$ and sends it to Bob.
4. Bob opens his commitment to Sk_{1-j}, c_{1-j} and sets $d = c_j$.
5. Alice verifies that $Dec_{ME}(K_0^{1-c_j} \oplus K_1^{1-c_j}, Sk_0) = 0$ (if $j = 1$) or that $Dec_{ME}(\hat{K}_0^{1-c_j} \oplus \hat{K}_1^{1-c_j}), Sk_1 = 0$ (if $j = 0$), otherwise she aborts the protocol.
6. Alice chooses $a_0, a_1 \in_R \{0,1\}$ and computes for $i \in \{0,1\}$: $\hat{a_i} = K_{a_i}^i \oplus (r_i \mid 0)\mathbf{G_0}$ if $j = 1$, if not, she computes $\hat{a_i} = \hat{K}_{a_i}^i \oplus (r_i \mid 0)\mathbf{G_1}$ where $r_i \in_R \{0,1\}^{k-1}$ and sends $(\hat{a_0}, \hat{a_1})$ to Bob.
7. Bob decrypts $\hat{a_d}$ to find a_d, where $Dec_{ME}(\hat{a_d}) = (r \mid a_d)$. If $Dec_{ME}(\hat{a_d})$ yields a decoding error, then Bob outputs $a_d = 0$.

Theorem 2. *Under the assumption that the bit commitment protocol is secure, Protocol 2 implements a randomized oblivious transfer that is complete and secure for Bob against active attacks according to Definition 1 under Assumptions 1 and 2. Furthermore, the probability that an actively cheating Bob learns both a_0 and a_1 is at most $\frac{1}{2} + \epsilon(n)$ where $\epsilon(n)$ is negligible.*

Proof. **Completeness:** an honest Bob passes the test of step 5 and receives a valid $\hat{a}_d$, so he can compute a_d.

Security for Alice: Bob must compute both $K_1^0 = K_1^1 = (r \mid 1)\mathbf{G_0} \oplus e$ or $\hat{K}_1^0 = \hat{K}_1^1 = (r \mid 1)\mathbf{G_1} \oplus e$ in order to learn both a_0 and a_1.

If Bobs computes (K_1^0, K_1^1) and $(\hat{K}_1^0, \hat{K}_1^1)$ honestly, *i.e*, according to the protocol, the probability that he learns both a_0 and a_1 is the same as in the Semi-Honest model protocol (negligible). If Bob computes both (K_1^0, K_1^1) and $(\hat{K}_1^0, \hat{K}_1^1)$ maliciously, then Alice will stop the protocol in step 5 with overwhelming probability and Bob won't learn a_0 nor a_1.

The best strategy for a rational malicious Bob is to compute one of the matrix pairs honestly and compute the other one maliciously, so that he learns both a_0 and a_1 if Alice asks him to open the commitment to the honestly computed matrix pair's secret key Sk_i, c_i. Note that Alice asks him to open the commitment to the maliciously computed

matrix pair's secret key with probability $\frac{1}{2}$. In this case, Bob will open the commitment to the expected honest matrix pair's secret key only with negligible probability. Thus, a malicious Bob learns both a_0 and a_1 with probability of at most $\frac{1}{2} + \epsilon(n)$ where $\epsilon(n)$ is negligible.

Security for Bob: Bob does not open his commitment to Sk_j, c_j, so the protocol is secure for Bob under Assumptions 1 and 2 analogously to Protocol 1. Alice would have to decrypt the received pair ((K_1^0, K_1^1) or $(\hat{K}_1^0, \hat{K}_1^1)$) or distinguish one of the ciphertexts in the pair from the other in order to learn d, but, since the randomized McEliece cryptosystem is IND-CPA secure [17], a PPT malicious Alice wouldn't be able to do so.

Differently from the Semi-Honest Model, a malicious Alice could also send an invalid value for $\hat{a_i}$ hoping that $i = d$ and expecting that Bob will complain when he encounters the decoding error. However, if Alice sends an invalid syndrome Bob will output 0 by default, as specified in Step 8 of this protocol, which is equivalent to the case where Alice sets her input $a_i = 0$, thwarting the attack.

It follows that the protocol is secure against an actively cheating Alice. □

4.2. Derandomizing Protocol 2

We now use the method presented by Beaver [1] to convert the randomized oblivious transfer implemented by Protocol 2 into "regular" oblivious transfer with the same level of security:

Protocol 3

1. Bob and Alice execute Protocol 2. Alice receives a_0, a_1 and Bob receives d, a_d.
2. Bob inputs c, sets $e = c \oplus d$ and sends e to Alice.
3. Alice inputs $b_0, b_1 \in \{0, 1\}$, computes $x_0 = b_0 \oplus a_e$ and $x_1 = b_1 \oplus a_{1 \oplus e}$ and sends x_0, x_1 to Bob.
4. Bob computes $b_c = x_c \oplus a_d$.

Theorem 3. *Protocol 3 implements oblivious transfer with the same level of security of Protocol 2.*

Proof. **Completeness:** An honest Bob, who knows a_d, can always recover b_c because $x_c = b_c \oplus a_{c \oplus e} = b_c \oplus a_d$.

Security for Alice: A malicious Bob can only recover both b_0 and b_1 if he knows both a_0 and a_1, because $x_{1 \oplus c} = b_{1 \oplus c} \oplus a_{1 \oplus c \oplus e} = b_{1 \oplus c} \oplus a_{1 \oplus d}$.

Security for Bob: A malicious Alice would have to discover d in order to compute c, thus the security for Bob follows from the security of Protocol 2.

□

4.3. A Fully Secure Protocol

In order to reduce the probability of Bob cheating in Protocol 9, we use the reduction of [5] to minimize the probability that a malicious Bob learns both of Alice inputs. Following this reduction, Protocol 3 is executed in parallel s times, where s is the security parameter. In each execution the inputs are chosen in such a way that Bob must learn both bits in all

executions in order to compute both Alice inputs in Protocol 4. The proof is identical to the one presented in [6].

Protocol 4

- Alice inputs $b_0, b_1 \in \{0,1\}$ and $b_{0,1}, b_{0,2}, \cdots, b_{0,s}, b_{1,1}, b_{1,2}, \cdots, b_{1,s}$.
- Bob inputs $c \in \{0,1\}$.
- Protocol 3 is executed s times, with inputs $b_{0,i}, b_{1,i}$ from Alice and $c_i = i$ from Bob for $i = 1 \cdots s$.
- Bob computes $b_c = b_{c,1} \oplus b_{c,2} \oplus \cdots \oplus b_{c,s}$.

Theorem 4. *Assuming that the bit commitment scheme used in Protocol 2 is secure, Protocol 4 is complete and secure for both Alice and Bob against active attacks according to Definition 1 under Assumptions 1 and 2.*

5. Conclusions

We provided, to the best of our knowledge, the first oblivious transfer protocol based on coding assumptions which has unconditional security for one of the parties. Our result follows from an observation we believe might be of independent interest: we note that the McEliece PKC is re-randomizable. Nice as it is, we payed a heavy price for our result. While previous constructions implement, without any further need of modification, string oblivious transfers, ours only works for bit oblivious transfer. To obtain efficient string code-based oblivious-transfer protocols seems to be an interesting sequel to this work.

References

[1] Donald Beaver. Precomputing oblivious transfer. In *CRYPTO '95: Proceedings of the 15th Annual International Cryptology Conference on Advances in Cryptology*, pages 97–109, London, UK, 1995. Springer-Verlag.

[2] Mihir Bellare and Silvio Micali. Non-interactive oblivious transfer and applications. In *CRYPTO '89: Proceedings on Advances in cryptology*, pages 547–557, New York, NY, USA, 1989. Springer-Verlag New York, Inc.

[3] Elwyn R Berlekamp, Robert McEliece, and Henk C A van Tilborg. On the inherent intractability of certain coding problems (corresp. *IEEE Transactions on Information Theory*, (24), 1978.

[4] Claude Crépeau, Jeroen van de Graaf, and Alain Tapp. Committed oblivious transfer and private multi-party computation. In Don Coppersmith, editor, *CRYPTO*, volume 963 of *Lecture Notes in Computer Science*, pages 110–123. Springer, 1995.

[5] Ivan Damgård, Joe Kilian, and Louis Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In *EUROCRYPT'99: Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, pages 56–73, Berlin, Heidelberg, 1999. Springer-Verlag.

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

[7] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In *CRYPTO*, pages 205–210, 1982.

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

[9] Jean-Bernard Fischer and Jacques Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In *EUROCRYPT'96: Proceedings of the 15th annual international conference on Theory and application of cryptographic techniques*, pages 245–255, Berlin, Heidelberg, 1996. Springer-Verlag.

[10] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229, New York, NY, USA, 1987. ACM.

[11] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In *In EUROCRYPT 2005, Springer-Verlag (LNCS 3494*, pages 78–95. Springer, 2005.

[12] Joe Kilian. Founding crytpography on oblivious transfer. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 20–31, New York, NY, USA, 1988. ACM.

[13] K. Kobara, K. Morozov, and R. Overbeck. Oblivious transfer via mceliece's pkc and permuted kernels. Cryptology ePrint Archive, Report 2007/382, 2007. `http://eprint.iacr.org/`.

[14] R J McEliece. A public-key cryptosystem based on algebraic coding theory. dsn progress report. In *Jet Propulsion Laboratories, CALTECH*, pages 42–44, 1978.

[15] R.J. McEliece. *The Theory of Information and Coding*, volume 3 of *The Encyclopedia of Mathematics and Its Applications*. Reading, Mass., Addison-Wesley, 1077.

[16] Moni Naor. Bit commitment using pseudo-randomness. In *CRYPTO '89: Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, pages 128–136, London, UK, 1990. Springer-Verlag.

[17] Ryo Nojima, Hideki Imai, Kazukuni Kobara, and Kirill Morozov. Semantic security for the mceliece cryptosystem without random oracles. *Des. Codes Cryptography*, 49(1-3):289–305, 2008.

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

[19] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 84–93, New York, NY, USA, 2005. ACM.