Sādhanā (2020)45:209
https://doi.org/10.1007/s12046-020-01447-6

1-out-of-2: post-quantum oblivious transfer protocols based on multivariate public key cryptography

NIBEDITA KUNDU[1], SUMIT KUMAR DEBNATH[2] and DHEERENDRA MISHRA[1,*]

[1]Department of Mathematics, The LNM Institute of Information Technology, Jaipur 302031, India
[2]Department of Mathematics, National Institute of Technology Jamshedpur, Jamshedpur 831014, India
e-mail: dheerendra.mishra@lnmiit.ac.in

MS received 23 January 2020; revised 26 May 2020; accepted 30 May 2020

Abstract. Oblivious transfer (OT) is a fundamental cryptographic primitive. It is developed for the efficient and feasible implementation of most advanced cryptographic tasks. Today, most of the existing OT protocols' security is based on number-theoretic assumptions. However, many number-theoretical problems are solvable by a quantum computer in polynomial time. Therefore, OT protocols with post-quantum cryptography approach are required. Multivariate cryptographic constructions are one of the potential candidates for post-quantum cryptography as they are speedy and require only modest computational resources. This paper presents constructions of OT protocols utilizing multivariate public key cryptography (MPKC). Security of our schemes is achieved under the hardness of multivariate quadratic (MQ) problem. To the best of our knowledge, our designs are the *first* MPKC-based post-quantum OT protocols.

Keywords. Multivariate public key cryptography; post-quantum cryptography; oblivious transfer; security.

1. Introduction

Oblivious transfer (OT) is one of the widely used modern cryptographic primitives. The concept of OT was introduced by Rabin [1]. It is a two-party cryptographic protocol, which allows a receiver to obliviously retrieve a single message of its choice from sender's database of messages so that the sender remains oblivious about the message received by the receiver, and receiver does not get any information about the other members of sender's database. Notably, an OT protocol is known as 1-out-of-n OT if the sender's database size is n and receiver gets only a single message of its choice. One can employ OT protocol as a building block in the constructions of secure multiparty computation (MPC) [2], private information retrieval [3], privacy-preserving keyword search [4], private and verifiable smart contracts, exchange of secrets, blockchain technology, signing contracts, etc. Due to the simplicity of functionality, most advanced cryptographic tasks can be implemented by it in an efficient way [5, 6]. However, in order to perform those tasks practically, we require a considerable number of OT executions. It makes the necessity of designing and secure OT protocols. Most of the existing OT protocols rely on the hardness of number-theoretic assumptions, such as hard discrete logarithm problem and hard factorization problem. However, these problems are not immune to quantum attack due to Shor's algorithm [7]. Thus, it is significant to construct a post-quantum secure OT protocol. The recent call of the NIST was mainly focused on post-quantum encryption and signatures. However, extending the post-quantum research into two-party protocols, particularly into OT, is an exciting direction of research as it can be combined with critical cryptographic tasks.

Apart from code-based, hash-based and lattice-based cryptosystems, multivariate cryptographic constructions are potential candidates for the post-quantum era. The security of multivariate public-key cryptography (MPKC) constructions relies on the assumption that solving a system of multivariate polynomial equations over a finite field is a hard problem. This problem is NP-hard [8, 9] because no known quantum algorithm can solve it in polynomial-time. The quadratic case of the problem is defined as the multivariate quadratic (MQ) problem. For the use of low-cost devices like smart cards and RFID chips [10, 11], MPKC-based schemes are attractive as they are speedy and require only modest computational resources. A public key for MPKC of degree two has the following form over the finite field:

*For correspondence
Published online: 19 August 2020

$$
\begin{aligned}
p_1(x_1,\ldots,x_n) &= \sum_{1\le i\le j\le n} \alpha_{1ij}x_ix_j + \sum_{1\le i\le n} \beta_{1i}x_i + \gamma_1\\
p_2(x_1,\ldots,x_n) &= \sum_{1\le i\le j\le n} \alpha_{2ij}x_ix_j + \sum_{1\le i\le n} \beta_{2i}x_i + \gamma_2\\
&\vdots\\
p_m(x_1,\ldots,x_n) &= \sum_{1\le i\le j\le n} \alpha_{mij}x_ix_j + \sum_{1\le i\le n} \beta_{mi}x_i + \gamma_m.
\end{aligned}
\tag{1.1}
$$

Considering the post-quantum security related issue, it is essential to design OT that remains secure against a quantum computer. Although there are several post-quantum OT protocols based on lattice, code and hash, design of MPKC-based OT remains still open. Moreover, MPKC-based constructions are very fast and require only modest computational resources. Thus the construction of OT protocol based on MPKC becomes an interesting direction of further research.

1.1 *Related works*

As far as we are aware, till now, there is no construction of MPKC-based OT protocol. However, there are several OT protocols based on other candidates of post-quantum cryptography, such as lattice, code, etc.

Dowsley *et al.* [12] presented the first OT based on McEliece assumption [13]. Later, several code-based OT protocols were developed in [14, 14, 15, 15–17] under the McEliece assumption. However, none of them achieve security in the UC framework. Rather, they achieve security in the stand-alone model. David *et al.* [18] proposed the first UC-secure code-based OT protocol under the McEliece assumption. However, the OT of [18] is inefficient in practice due to the use of some explicit cryptographic primitives.

Peikert *et al.* [19] developed a simple and general framework for constructing OT protocols based on LWE. However, their construction is inefficient due to the use of post-quantum public-key encryption. Following this, a sequence of lattice-based OT protocols were presented in [20–23]. Liu ad Hu [24] designed improved UC-secure OT protocols based on the LWE and RLWE assumptions. Later, Branco *et al.* [25] claimed that the scheme [24] does not provide security for the receiver and proposed an RLWE-based OT protocol. Blazy *et al.* [26] developed a UC-secure OT protocol that deals with adaptive corruptions under the LWE assumption.

Recently, Barreto *et al.* [27] implemented an functionality combining the OT scheme of Chou and Orlandi [28] with the supersingular isogeny Diffie–Hellman (SIDH) primitive of Feo *et al.* [29]. More recently, Branco *et al.* [30] developed an efficient and versatile framework for OT protocol utilizing one-round key-exchange (ORKE).

1.2 *Our contribution*

To date, most of the OT protocols are based on the classical number-theoretic assumptions such as solving discrete logarithms and factorizing large integers. These problems are assumed to be hard for traditional algorithms. However, one can efficiently break the security of these problems using a quantum computer due to Shor's algorithm. Thus, as soon as the efficient quantum computer comes into the market, the safety of classical OT protocols will immediately break down. This brings the necessity of OT protocols, which can resist future attacks of quantum computers.

In the literature, there are several post-quantum OT protocols based on lattice, code, etc. However, to date, there is no construction of multivariate cryptography-based OT protocol. Moreover, multivariate cryptography-based constructions are a potential candidate for post-quantum cryptography as they are speedy and require only modest computational resources. These motivate us to design OT protocols based on MPKC.

In this paper, we first develop two 1-out-of-2 OT protocols, namely OT1 and OT2, by employing an MPKC encryption scheme. Any MPKC encryption scheme can be employed as the building blocks of our schemes. Security of OT1 and OT2 is given in the semi-honest model. We then extend these OT1 and OT2 to OT3 and OT4, respectively, by employing a 5-pass identification protocol of [31] to attain security in the presence of the malicious server. In particular, our designs are the *first* multivariate cryptography-based OT protocols. Our protocols are quantum computer resistant (i.e., post-quantum) under the hardness of MQ problem. Consider the underlying field $\mathbb{F}_q$ as the binary field, the message size as n bits and the ciphertext size as approximately n bits. Then the communication complexity of our OT1 is approximately $n^3 + 3n^2 + 4n$ bits and computation complexity is approximately $(n^3 + n^2)$-bit multiplication along with the cost of one MPKC decryption. On the other side, the communication complexity of OT2 is approximately $\frac{(3n^3+9n^2+10n)}{2}$ bits and the computation complexity is approximately $(n^3 + n^2)$-bit multiplication along with the cost of one MPKC decryption. Both the OT1 and OT2 require only 3 rounds to complete the process of OT. Note that the communication and the computation complexity of OT3 and OT4 will be increased due to the addition of the 5-pass identification protocol. Moreover, if the non-interactive form of the identification protocol is

used then the round complexity of each of OT3 and OT4 will remain the same, i.e. 3, but the underlying security model will be random oracle model instead of the standard, unlike OT1 and OT2.

2. Preliminaries

Basic notations:

Throughout the paper, $x \in_R A$ represents that x is chosen uniformly at random from a set A, $\perp$ denotes "nothing", $\mathbb{F}_q$ stands for finite field of order q, $\mathbb{F}_{q^n}$ denotes an n degree extension field of $\mathbb{F}_q$ and $(\mathbb{F}_q)^n$ represents $\{\mathbf{x} = (x_1, \ldots, x_n) | x_i \in \mathbb{F}_q \text{ for } i = 1, \ldots, n\}$.

OT functionality: In this work, we realize the OT functionality $F_{\mathsf{OT}} : (\{\mathbf{x}_0, \mathbf{x}_1\}, \sigma) \rightarrow (\perp, \mathbf{x}_\sigma)$, where $\{\mathbf{x}_0, \mathbf{x}_1\}$ is the input of the sender, $\sigma \in \{0, 1\}$ is the input of the receiver, $\perp$ is the output of the sender and $\mathbf{x}_\sigma$ is the output of the receiver.

2.1 *Hardness assumption*

MQ assumption [32]: Given a system of m multivariate quadratic polynomials $\{p_1(x_1, \ldots, x_n), \ldots, p_m(x_1, \ldots, x_n)\}$ in n variables $x_1, \ldots, x_n$ over a finite filed $\mathbb{F}_q$ as shown in 1.1, it is hard to find a solution $\mathbf{x} = (x_1, \ldots, x_n)$ of the system of equations $p_1(\mathbf{x}) = \ldots, p_m(\mathbf{x}) = 0$.

2.2 *General construction of MPKC encryption [33]*

It consists of three algorithms: (i) ME.KGen, (ii) ME.Enc and (iii) ME.Dec.

$(pk, sk) \leftarrow \mathsf{ME.KGen}(1^\kappa)$: On input 1^κ, this algorithm generates a secret key $sk = (S, F, T)$ and a public key $pk = P = S \circ F \circ T$, where $F : (\mathbb{F}_q)^n \rightarrow (\mathbb{F}_q)^m$ is a central map and $S : (\mathbb{F}_q)^m \rightarrow (\mathbb{F}_q)^m$, $T : (\mathbb{F}_q)^n \rightarrow (\mathbb{F}_q)^n$ are invertible affine transformations with $m(\kappa) \geq n(\kappa)$.

$(\mathbf{y}) \leftarrow \mathsf{ME.Enc}(pk, \mathbf{x})$: Given a message $\mathbf{x} \in (\mathbb{F}_q)^n$ and a public key $pk = P = S \circ F \circ T$, the encryptor derives the ciphertext $\mathbf{y} = P(\mathbf{x}) \in (\mathbb{F}_q)^m$.

$(\mathbf{x}) \leftarrow \mathsf{ME.Dec}(sk, \mathbf{y})$: To decrypt the ciphertext $\mathbf{y} \in (\mathbb{F}_q)^m$ using the secret key $sk = (S, F, T)$, decryptor recursively calculates $\mathbf{z} = S^{-1}(\mathbf{y}) \in (\mathbb{F}_q)^m$, $\mathbf{w} = F^{-1}(\mathbf{z}) \in (\mathbb{F}_q)^n$ and $\mathbf{x} = T^{-1}(\mathbf{w}) \in (\mathbb{F}_q)^n$. Finally, it outputs $\mathbf{x} \in (\mathbb{F}_q)^n$ as the plaintext corresponding to the ciphertext $\mathbf{y} \in (\mathbb{F}_q)^m$. Note that computing F^{-1} means finding a pre-image under F.

2.3 *Five-pass identification protocol [31]*

The 5-pass identification protocol is a zero-knowledge argument of knowledge for a solution of the system $\mathcal{F}(\mathbf{x}) = \mathbf{v}$, where $\mathcal{F} : (\mathbb{F}_q)^n \rightarrow (\mathbb{F}_q)^m$ is a system of $m = m(\kappa)$ multivariate quadratic polynomials in $n = n(\kappa)$ variables, κ being the security parameter. Let $\mathcal{G}$ be the polar form of $\mathcal{F}$. Then $\mathcal{G}(\mathbf{x}, \mathbf{y}) = \mathcal{F}(\mathbf{x} + \mathbf{y}) - \mathcal{F}(\mathbf{x}) - \mathcal{F}(\mathbf{y}) + \mathcal{F}(\mathbf{0})$. Note that $\mathcal{G}$ is a bilinear mapping. Let the prover Prov have the solution $\mathbf{s}$ of the system $\mathcal{F}(\mathbf{x}) = \mathbf{v}$, i.e. $\mathcal{F}(\mathbf{s}) = \mathbf{v}$, and *Comit* be a commitment scheme that satisfies statistically hiding property and computationally binding property. We denote the 5-pass identification protocol by $\mathsf{IdP}\{\mathbf{s} | \mathcal{F}(\mathbf{s}) = \mathbf{v}\}$, where Prov wants to convince the verifier Ver that it has the knowledge of a solution $\mathbf{s}$ of the system $\mathcal{F}(\mathbf{x}) = \mathbf{v}$, without revealing the secret $\mathbf{s}$ to the verifier. Note that the verifier has the knowledge of $\mathbf{v}$ and the public parameter $\mathcal{F}$. The prover Prov, in order to prove its identity $\mathbf{s}$, performs five rounds with the verifier Ver as depicted in Fig 1. Note that the interactive 5-pass identification protocol can be transformed into non-interactive version with the help of the Fiat–Shamir technique [34]. Security of the 5-pass protocol depends on the hardness of MQ problem, provided the underlying commitment scheme *Comit* is computationally binding and statistically hiding. If *Comit* is statistically hiding then the 5-pass identification protocol is statistically zero knowledge. On the other side, if *Comit* is computationally binding then the 5-pass identification protocol is argument of knowledge with knowledge error $1/2 + 1/2q$.

Argument of knowledge: This property ensures that a cheating prover Prov cannot generate a valid proof of the system $\mathcal{F}(\mathbf{x}) = \mathbf{v}$ with the knowledge of the secret $\mathbf{s}$. We will show that if a cheating prover Prov can generate a valid proof of the system $\mathcal{F}(\mathbf{x}) = \mathbf{v}$ with the knowledge of $\mathbf{s}$ then a knowledge extractor can be constructed to extract the knowledge of the secret $\mathbf{s}$. The knowledge extractor generates four valid transcripts $((c_0, c_1), a^{(i)}, t_1^{(i)}, e_1^{(i)}, chl^{(j)}, \mathsf{Re}^{(i,j)})$ for $i, j \in \{0, 1\}$ from the prover such that $a^{(0)} \neq a^{(1)}$ and $chl^{(j)} = j$. Note that

$$c_0 = \mathsf{Comit}\left(\mathbf{r}_0^{(0)}, a^{(0)}\mathbf{r}_0^{(0)} - \mathbf{t}_1^{(0)}, a^{(0)}\mathcal{F}(\mathbf{r}_0^{(0)}) - \mathbf{e}_1^{(0)}\right) = \mathsf{Comit}\left(\mathbf{r}_0^{(1)}, a^{(1)}\mathbf{r}_0^{(1)} - \mathbf{t}_1^{(1)}, a^{(1)}\mathcal{F}(\mathbf{r}_0^{(1)}) - \mathbf{e}_1^{(1)}\right) \tag{2.1}$$

$$c_1 = \mathsf{Comit}\left(\mathbf{r}_1^{(0)}, a^{(0)}(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) - \mathcal{G}(\mathbf{t}_1^{(0)}, \mathbf{r}_1^{(0)}) - \mathbf{e}_1^{(0)}\right) = \mathsf{Comit}\left(\mathbf{r}_1^{(1)}, a^{(1)}(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(1)}) + \mathcal{F}(\mathbf{0})) - \mathcal{G}(\mathbf{t}_1^{(1)}, \mathbf{r}_1^{(1)}) - \mathbf{e}_1^{(1)}\right) \tag{2.2}$$

If the arguments of the Comit are distinct in 2.1 then the binding property of Comit is broken. Similarly, if the arguments of the Comit are distinct in 2.2 then the binding property of Comit is broken. Thus, we have

$$\mathbf{r}_0^{(0)} = \mathbf{r}_0^{(1)} \tag{2.3}$$

$\mathsf{Prov}(\mathcal{F}, \mathbf{v}, \mathbf{s})$		$\mathsf{Ver}(\mathcal{F}, \mathbf{v})$
$\mathbf{r}_0, \mathbf{t}_0 \in_R (\mathbb{F}_q)^n, \mathbf{e}_0 \in_R (\mathbb{F}_q)^m$		
$\mathbf{r}_1 = \mathbf{s} - \mathbf{r}_0$		
$c_0 = \mathsf{Comit}(\mathbf{r}_0, \mathbf{t}_0, \mathbf{e}_0)$		
$c_1 = \mathsf{Comit}(\mathbf{r}_1, \mathcal{G}(\mathbf{t}_0, \mathbf{r}_1) + \mathbf{e}_0)$	$\xrightarrow{c_0, c_1}$	
	$\xleftarrow{a}$	$a \in_R \mathbb{F}_q$
$\mathbf{t}_1 = a\mathbf{r}_0 - \mathbf{t}_0$		
$\mathbf{e}_1 = a\mathcal{F}(\mathbf{r}_0) - \mathbf{e}_0$	$\xrightarrow{\mathbf{t}_1, \mathbf{e}_1}$	
	$\xleftarrow{chl}$	$chl \in_R \{0, 1\}$
If $chl = 0$, $\mathsf{Re} = \mathbf{r}_0$		
If $chl = 1$, $\mathsf{Re} = \mathbf{r}_1$	$\xrightarrow{\mathsf{Re}}$	
		If $chl = 0$ check
		$c_0 \stackrel{?}{=} \mathsf{Comit}(\mathbf{r}_0, a\mathbf{r}_0 - \mathbf{t}_1, a\mathcal{F}(\mathbf{r}_0) - \mathbf{e}_1)$
		If $chl = 1$ check
		$c_1 \stackrel{?}{=} \mathsf{Comit}(\mathbf{r}_1, a(\mathbf{v} - \mathcal{F}(\mathbf{r}_1) + \mathcal{F}(\mathbf{0})) - \mathcal{G}(\mathbf{t}_1, \mathbf{r}_1) - \mathbf{e}_1)$

Figure 1. 5-pass identification protocol.

$$a^{(0)}\mathbf{r}_0^{(0)} - \mathbf{t}_1^{(0)} = a^{(1)}\mathbf{r}_0^{(1)} - \mathbf{t}_1^{(1)} \tag{2.4}$$

$$a^{(0)}\mathcal{F}(\mathbf{r}_0^{(0)}) - \mathbf{e}_1^{(0)} = a^{(1)}\mathcal{F}(\mathbf{r}_0^{(1)}) - \mathbf{e}_1^{(1)} \tag{2.5}$$

$$\mathbf{r}_1^{(0)} = \mathbf{r}_1^{(1)} \tag{2.6}$$

$$\begin{aligned} &a^{(0)}(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) - \mathcal{G}(\mathbf{t}_1^{(0)}, \mathbf{r}_1^{(0)}) - \mathbf{e}_1^{(0)} \\ &= a^{(1)}(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(1)}) + \mathcal{F}(\mathbf{0})) - \mathcal{G}(\mathbf{t}_1^{(1)}, \mathbf{r}_1^{(1)}) - \mathbf{e}_1^{(1)}. \end{aligned} \tag{2.7}$$

From equations (2.6) and (2.7)

$$\begin{aligned} &(a^{(0)} - a_\rho^{(1)})(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) = \mathcal{G}(\mathbf{t}_1^{(0)}, \mathbf{r}_1^{(0)}) \\ &\quad - \mathcal{G}(\mathbf{t}_1^{(1)}, \mathbf{r}_1^{(1)}) + \mathbf{e}_1^{(0)} - \mathbf{e}_1^{(1)} \\ &\Rightarrow (a^{(0)} - a^{(1)})(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) \\ &= \mathcal{G}(\mathbf{t}_1^{(0)} - \mathbf{t}_1^{(1)}, \mathbf{r}_1^{(0)}) + \mathbf{e}_1^{(0)} - \mathbf{e}_1^{(1)}. \end{aligned} \tag{2.8}$$

From (2.3), (2.4), (2.5) and (2.8)

$$\begin{aligned} &(a^{(0)} - a^{(1)})(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) \\ &\quad = \mathcal{G}((a^{(0)} - a^{(1)})\mathbf{r}_0^{(0)}, \mathbf{r}_1^{(0)}) + (a^{(0)} - a^{(1)})\mathcal{F}(\mathbf{r}_0^{(0)}) \\ &\Rightarrow (a^{(0)} - a^{(1)})(\mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0})) \\ &\quad = (a^{(0)} - a^{(1)})\mathcal{G}(\mathbf{r}_0^{(0)}, \mathbf{r}_1^{(0)}) + (a^{(0)} - a^{(1)})\mathcal{F}(\mathbf{r}_0^{(0)}) \\ &\Rightarrow \mathbf{v} - \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{0}) = \mathcal{G}(\mathbf{r}_0^{(0)}, \mathbf{r}_1^{(0)}) \\ &\quad + \mathcal{F}(\mathbf{r}_0^{(0)}) \text{ since } a^{(0)} \neq a^{(1)} \\ &\Rightarrow \mathbf{v} = \mathcal{F}(\mathbf{r}_1^{(0)}) + \mathcal{G}(\mathbf{r}_0^{(0)}, \mathbf{r}_1^{(0)}) + \mathcal{F}(\mathbf{r}_0^{(0)}) - \mathcal{F}(\mathbf{0}) \\ &= \mathcal{F}(\mathbf{r}_0^{(0)} + \mathbf{r}_1^{(0)}). \end{aligned} \tag{2.9}$$

Thus the extractor extracts a solution $\mathbf{r}_0^{(0)} + \mathbf{r}_1^{(0)}$ of $\mathcal{F}(\mathbf{x}) = \mathbf{v}$.

3. OT protocols

In our constructions, addition and subtraction of two system of multivariate polynomials $P = (p_1, \ldots, p_m) : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $R = (r_1, \ldots, r_m) : \mathbb{F}_q^n \to \mathbb{F}_q^m$ will be denoted by $P \pm R$ and defined as $P \pm R(\mathbf{x}) = P(\mathbf{x}) \pm R(\mathbf{x}) = (p_1(\mathbf{x}) \pm r_1(\mathbf{x}), \ldots, p_m(\mathbf{x}) \pm r_m(\mathbf{x}))$.

3.1 *OT1*

It involves two participants: sender Sen with input $\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{F}_q^m$ and receiver Rec with input $\sigma \in \{0, 1\}$. A high-level overview is provided in Fig 2. Sender Sen first generates a public key-secret key pair using ME.KGen and sends the public key $R : \mathbb{F}_q^n \to \mathbb{F}_q^m$ to Rec. Then the receiver Rec runs the algorithm ME.KGen to generate public key-secret key pair $(P, (S_2, F_2, T_2))$ and sends $\{P_0\}$ to Sen, where $P_0 = P$ if $\sigma = 0$ and $P_0 = R - P_0$ if $\sigma = 1$. The sender Sen then computes $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ for its input $\{\mathbf{x}_0, \mathbf{x}_1\}$ and sends the resulting values to Rec, who in turn decrypts $P_\sigma(\mathbf{x}_\sigma)$ using the secret key (S_2, F_2, T_2) to extract $\mathbf{x}_\sigma$. The execution between the sender and the receiver is given below in detail:

Protocol 1 OT1

1. The sender Sen runs the algorithm ME.KGen to generate secret key (S_1, F_1, T_1) and public key $R = S_1 \circ F_1 \circ T_1 : \mathbb{F}_q^n \to \mathbb{F}_q^m$. It then sends R to Rec.
2. On receiving R from Sen, the receiver Rec runs the ME.KGen algorithm to generate secret key (S_2, F_2, T_2) and public key $P = S_2 \circ F_2 \circ T_2 : \mathbb{F}_q^n \to \mathbb{F}_q^m$. It then sets $P_\sigma = P$ and $P_{1-\sigma} = R - P_\sigma$ according to its choice $\sigma \in \{0, 1\}$. In the following, Rec forwards $\{P_0\}$ to Sen.
3. The sender Sen with input $\{\mathbf{x}_0, \mathbf{x}_1\}$, on receiving $\{P_0\}$ from Rec, computes $P_1 = R - P_0$, $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ and sends $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ to Rec.
4. On receiving $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ from Sen, the receiver Rec extracts $\mathbf{x}_\sigma$ by decrypting $P_\sigma(\mathbf{x}_\sigma)$ using the secret key (S_2, F_2, T_2). Note that $P_\sigma = P$.

3.2 *OT2*

Similar to OT1, it involves two participants: sender Sen with input $\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{F}_q^m$ and receiver Rec with input $\sigma \in \{0, 1\}$. See Fig 3 for a high-level overview of this protocol. Sender Sen sends two public key maps P_0, P_1 to Rec after generating them using ME.KGen. The receiver Rec generates secret key

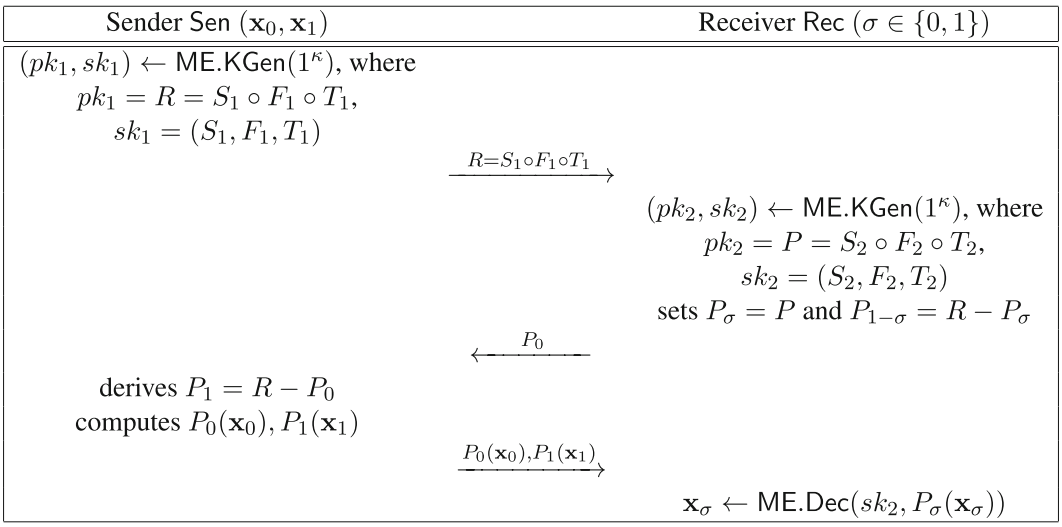

Figure 2. Interaction between the parties in OT1.

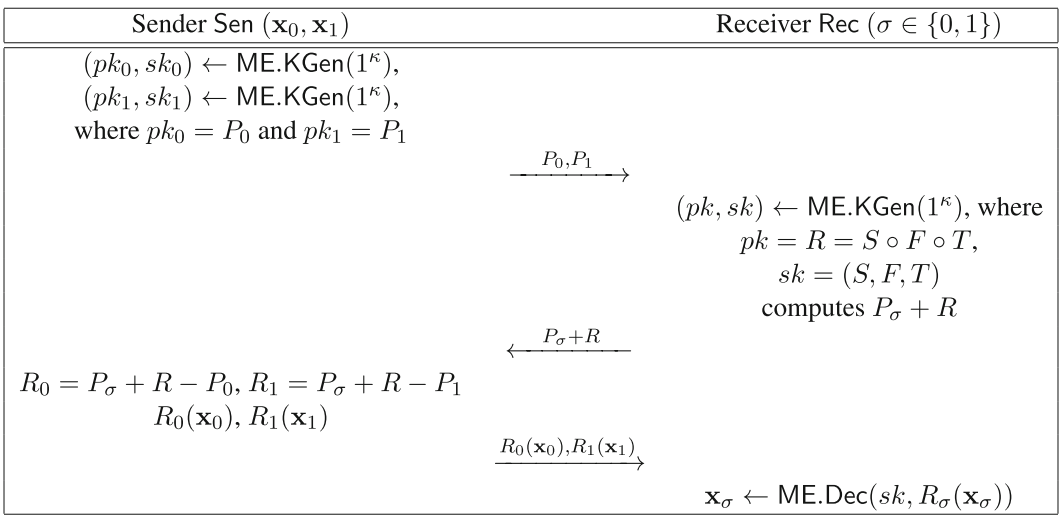

Figure 3. Interaction between the parties in OT2.

(S, F, T) and public key $R = S \circ F \circ T$ using the algorithm ME.KGen and sends $P_\sigma + R$ to Sen, who in turn sends $R_0(\mathbf{x}_0)$ and $R_1(\mathbf{x}_1)$ to Rec, where $R_0 = P_\sigma + R - P_0$ and $R_1 = P_\sigma + R - P_1$. The receiver Rec then decrypts $R_\sigma(\mathbf{x}_\sigma)$ using the secret key (S, F, T) to extract $\mathbf{x}_\sigma$ since $R_\sigma = R$. We describe here the execution between Sen and Rec:

Protocol 2 OT2

1. The sender Sen generates two public key maps P_0, P_1 from $\mathbb{F}_q^n$ to $\mathbb{F}_q^m$ by running the algorithm ME.KGen and sends P_0, P_1 to Rec.
2. The receiver Rec, on receiving P_0, P_1 from Sen, runs the algorithm ME.KGen to generate secret key (S, F, T) and public key $R = S \circ F \circ T : \mathbb{F}_q^n \to \mathbb{F}_q^m$. It then computes $P_\sigma + R$ for its choice $\sigma \in \{0, 1\}$ and sends $P_\sigma + R$ to Sen.
3. On receiving $P_\sigma + R$ from Rec, the sender Sen computes $R_0 = P_\sigma + R - P_0$ and $R_1 = P_\sigma + R - P_1$. It then determines $R_0(\mathbf{x}_0)$ and $R_1(\mathbf{x}_1)$ for its input $\{\mathbf{x}_0, \mathbf{x}_1\}$ and sends $R_0(\mathbf{x}_0)$, $R_1(\mathbf{x}_1)$ to Rec.
4. The receiver Rec, on receiving $R_0(\mathbf{x}_0)$, $R_1(\mathbf{x}_1)$ from Sen, decrypts $R_\sigma(\mathbf{x}_\sigma)$ using the secret key (S, F, T) to extract $\mathbf{x}_\sigma$. Note that $R_\sigma = R$.

3.3 *OT3*

The protocol OT3 is an extension of OT1, where the security is achieved against malicious sender instead of semi-honest sender utilizing the 5-pass identification protocol of [31]. A high-level overview of this protocol is depicted in Fig 4. The execution between the participants is discussed here:

Protocol 3 OT3

1. The sender Sen generates secret key (S_1, F_1, T_1) and public key $R = S_1 \circ F_1 \circ T_1 : \mathbb{F}_q^n \to \mathbb{F}_q^m$ using the algorithm ME.KGen, and sends R to Rec.
2. The receiver Rec then generates secret key (S_2, F_2, T_2) and public key $P = S_2 \circ F_2 \circ T_2 : \mathbb{F}_q^n \to \mathbb{F}_q^m$ by running the algorithm ME.KGen. It then sets $P_\sigma = P$ and $P_{1-\sigma} = R - P_\sigma$ based on the input $\sigma \in \{0, 1\}$, and forwards $\{P_0\}$ to Sen.
3. The sender Sen computes $P_1 = R - P_0$ and $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ with the help of its input with input $\{\mathbf{x}_0, \mathbf{x}_1\}$ and the public key $\{P_0\}$. In the following, it generates a proof $\mathsf{IdP}\{\mathbf{x}_0, \mathbf{x}_1 | P_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge P_1(\mathbf{x}_1) = \mathbf{w}_1\}$ and sends this proof along with $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ to Rec.
4. The receiver Rec checks the correctness of the proof $\mathsf{IdP}\{\mathbf{x}_0, \mathbf{x}_1 | P_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge P_1(\mathbf{x}_1) = \mathbf{w}_1\}$. If the proof is not valid it outputs $\perp$ and aborts; otherwise, it decrypts $P_\sigma(\mathbf{x}_\sigma)$ using the secret key (S_2, F_2, T_2) and outputs $\mathbf{x}_\sigma$. Note that $P_\sigma = P$.

3.4 *OT4*

The protocol OT2 is extended to OT4 in order to gain security in the presence of malicious sender utilizing the 5-pass identification protocol of [31]. See Fig 5 for a high-

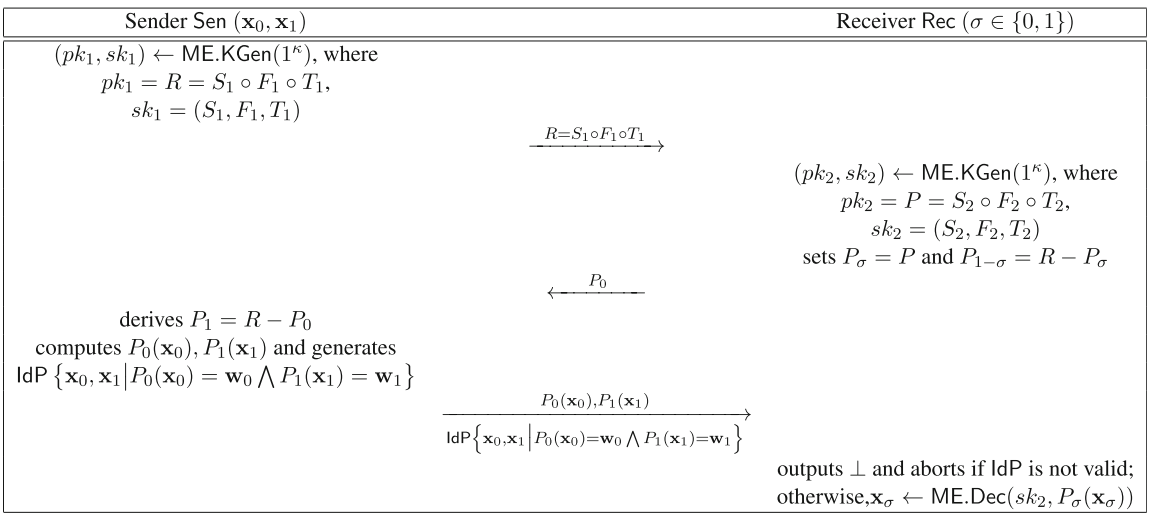

Figure 4. Interaction between the parties in OT3.

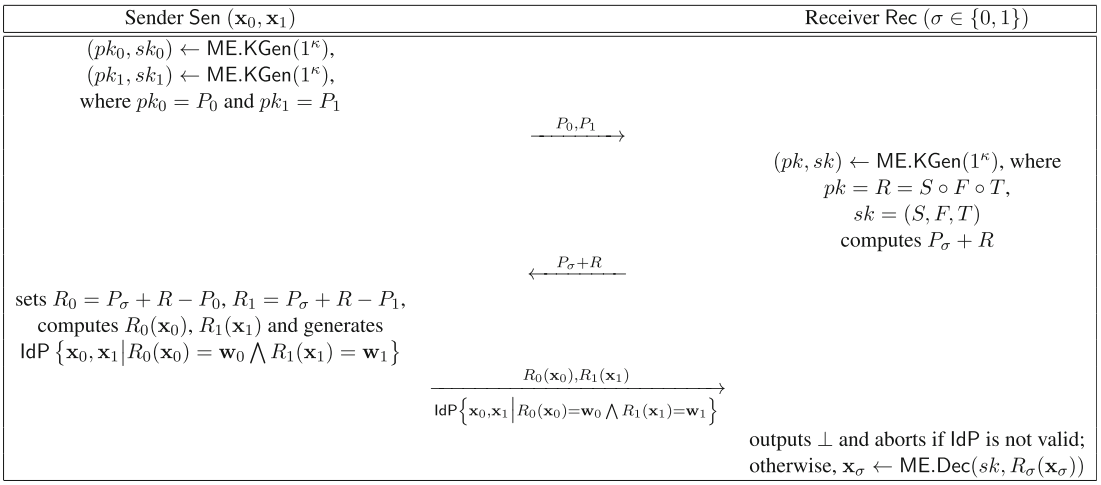

Figure 5. Interaction between the parties in OT4.

level overview of OT4. We discuss here the execution between the participants.

Protocol 4 OT4

1. The sender Sen sends two public key maps P_0, P_1 to Rec after generating them using the algorithm ME.KGen.
2. On receiving P_0, P_1 from Sen, the receiver Rec generates secret key (S, F, T) and public key $R = S \circ F \circ T : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^m$ by running the algorithm ME.KGen. It then computes $P_\sigma + R$ for its input $\sigma \in \{0, 1\}$ and sends $P_\sigma + R$ to Sen.
3. The sender Sen then executes $R_0 = P_\sigma + R - P_0$ and $R_1 = P_\sigma + R - P_1$, and determines $R_0(\mathbf{x}_0)$ and $R_1(\mathbf{x}_1)$ for its input $\{\mathbf{x}_0, \mathbf{x}_1\}$. It then generates a proof $\mathsf{IdP}\left\{\mathbf{x}_0, \mathbf{x}_1 \middle| R_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge R_1(\mathbf{x}_1) = \mathbf{w}_1\right\}$ and sends the proof along with $R_0(\mathbf{x}_0)$, $R_1(\mathbf{x}_1)$ to Rec.
4. The receiver Rec checks the correctness of the proof $\mathsf{IdP}\left\{\mathbf{x}_0, \mathbf{x}_1 \middle| R_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge R_1(\mathbf{x}_1) = \mathbf{w}_1\right\}$. If the verification fails, then Rec outputs $\perp$ and aborts; otherwise, it decrypts $R_\sigma(\mathbf{x}_\sigma)$ using the secret key (S, F, T) and outputs $\mathbf{x}_\sigma$. Note that $R_\sigma = R$.

4. Security

Theorem 4.1 *If the MQ assumption holds, then OT1 securely computes the functionality* $F_{\mathsf{OT1}} : (\{\mathbf{x}_0, \mathbf{x}_1\}, \sigma) \rightarrow (\perp, \mathbf{x}_\sigma)$ *against semi-honest adversaries.*

Proof Correctness: If $\sigma = 0$ then $P_\sigma = P_0 = P$. This implies $P_1 = R - P$. Therefore, the decryption of $P_\sigma(x_\sigma)$, i.e. $P_0(x_0)$, i.e. $P(x_0)$, using sk_2 will produce x_0, i.e. x_σ. On the other hand, if $\sigma = 1$ then $P_\sigma = P_1 = P$, which implies $P_0 = R - P$. Thus the decryption of $P_\sigma(x_\sigma)$, i.e. $P_1(x_1)$, i.e. $P(x_1)$, using sk_2 will produce x_1, i.e. x_σ.

Security of OT1 is proved by considering two cases. In the first case the sender Sen is considered as corrupted, while in the second case the receiver Rec is considered as corrupted. In each of the cases, a simulator *SIM*, with access to the corrupted party's input and output, will be constructed to simulate the OT1 protocol such that the simulated view and the real world view are indistinguishable. Note that a participant's view consists of input messages of the participant, the outcome of the participant's internal coin tosses and the messages received by the participant during the protocol execution.

Case I (Sender Sen *is corrupted):* Let *SIM* have access to the sender Sen's input $(\mathbf{x}_0, \mathbf{x}_1)$ and output $\perp$. Then *SIM* executes the following:

– runs the algorithm ME.KGen on input 1^κ to generate public key P_0'
– uniformly chooses its random coins $t^{(1)}$ and outputs its view as $(\{\mathbf{x}_0, \mathbf{x}_1\}; t^{(1)}; P_0')$.

The real view contains $\{\mathbf{x}_0, \mathbf{x}_1\}$, random coins and P_0. Note that the input $\{\mathbf{x}_0, \mathbf{x}_1\}$ is the same in both the views. Since the outcome of internal random coins $t^{(1)}$ is uniformly random, its distribution is the same as in a real execution. Moreover, the distribution of $\{P_0'\}$ is indistinguishable from the distribution of $\{P_0\}$ since the invertible affine transformations (S, T) and the associated central map (F) are randomly chosen at the time of running the key generation algorithm ME.KGen. Thus the simulated view $(\{\mathbf{x}_0, \mathbf{x}_1\}; t^{(1)}; P_0')$ is indistinguishable from the view in a real execution.

Case II (Receiver Rec *is corrupted):* Assume that the simulator *SIM* has access to the receiver Rec's input $\sigma \in \{0, 1\}$ and output $\mathbf{x}_\sigma$. Then it performs the following steps:

– runs the algorithm ME.KGen on input 1^κ to generate public keys R', P' and sets $P'_\sigma = P'$, $P'_{1-\sigma} = R' - P'_\sigma$
– randomly selects $\mathbf{x}'$ from $\mathbb{F}_q^n$ and computes $P'_0(\mathbf{x}'_0), P'_1(\mathbf{x}'_1)$, where $\mathbf{x}'_\sigma = \mathbf{x}_\sigma$ and $\mathbf{x}'_{1-\sigma} = \mathbf{x}'$
– uniformly chooses its random coins $t^{(2)}$ and outputs its view as $(\sigma; t^{(2)}; R', P'_0(\mathbf{x}'_0), P'_1(\mathbf{x}'_1))$.

The real view consists of σ, the random coins and $R, P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)$. In both the views, the input σ is the same and the distribution of random coins is also the same since the outcome of internal random coins $t^{(2)}$ is uniformly random. Note that the distribution of $\{R'\}$ is indistinguishable from the distribution of $\{R\}$ since the invertible linear transformations (S, T) and the associated central map (F) are randomly selected during running the key generation algorithm ME.KGen. Furthermore, the distribution of $\{P'_0(\mathbf{x}'_0), P'_1(\mathbf{x}'_1)\}$ is indistinguishable from that of $\{P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)\}$ under the hardness of MQ assumption. Thus, the real view and the simulated view are indistinguishable. □

Theorem 4.2 *The OT2 securely computes the functionality* $F_{\mathsf{OT2}} : (\{\mathbf{x}_0, \mathbf{x}_1\}, \sigma) \rightarrow (\perp, \mathbf{x}_\sigma)$ *against semi-honest adversaries under the hardness of MQ assumption.*

Proof *Correctness:* Let us consider $\sigma = 0$. Then $P_\sigma = P_0$, which implies $R_0 = R$ and $R_1 = P_0 + R - P_1$. Hence, the decryption of $R_\sigma(x_\sigma)$, i.e. $R_0(x_0)$, i.e. $R(x_0)$, using sk will produce x_0, i.e. x_σ. On the other hand, if $\sigma = 1$ then $P_\sigma = P_1$. This implies $R_0 = P_1 + R - P_0$ and $R_1 = R$. Therefore, the decryption of $R_\sigma(x_\sigma)$, i.e. $R_1(x_1)$, i.e. $R(x_1)$, using sk will produce x_1, i.e. x_σ.

Similar to OT1, the security of OT2 can be proved by considering the following cases.

Case I (Sender Sen *is corrupted):* Consider a simulator *SIM* with access to the sender Sen's input $\{\mathbf{x}_0, \mathbf{x}_1\}$ and output $\perp$. The simulator *SIM* proceeds as follows:

– generates a public key R' by running the algorithm ME.KGen on input 1^κ
– uniformly chooses its random coins $t^{(1)}$ and outputs its view as $(\{\mathbf{x}_0, \mathbf{x}_1\}; t^{(1)}; R')$.

The real view consists of $\{\mathbf{x}_0, \mathbf{x}_1\}$, random coins and R. The input $\{\mathbf{x}_0, \mathbf{x}_1\}$ is the same in both the views. Moreover the distribution of the random coins is the same as the outcome of internal random coins $t^{(1)}$, which is uniformly random. Furthermore, the distribution of $\{R'\}$ is indistinguishable from the distribution of $\{R\}$ since the invertible linear transformations (S, T) and the associated central map (F) are randomly selected during running the key generation algorithm ME.KGen. Therefore, the simulated view $(\{\mathbf{x}_0, \mathbf{x}_1\}; t^{(1)}; R')$ is indistinguishable from the real view.

Case II (Receiver Rec *is corrupted):* Let us assume that the simulator *SIM* has access to the receiver Rec's input $\sigma \in \{0, 1\}$ and output $\mathbf{x}_\sigma$. Then the simulator *SIM* does the following:

– generates public keys R', P'_0, P'_1 by running the algorithm ME.KGen on input 1^κ and sets $R'_\sigma = R'$ and $R'_{1-\sigma} = P'_\sigma + R' - P'_{1-\sigma}$
– randomly selects $\mathbf{x}'$ from $\mathbb{F}_q^n$ and computes $R'_0(\mathbf{x}'_0), R'_1(\mathbf{x}'_1)$, where $\mathbf{x}'_\sigma = \mathbf{x}_\sigma$ and $\mathbf{x}'_{1-\sigma} = \mathbf{x}'$
– uniformly chooses its random coins $t^{(2)}$ and outputs its view as $(\sigma; t^{(2)}; P'_0, P'_1, R'_0(\mathbf{x}'_0), R'_1(\mathbf{x}'_1))$.

The real view contains σ, the random coins and $P_0, P_1, R_0(\mathbf{x}_0), R_1(\mathbf{x}_1)$. In both the views, the input σ is the same and the distribution of random coins is also the same since the outcome of internal random coins $t^{(2)}$ is uniformly random. Moreover, the distribution of $\{P'_0, P'_1\}$ is indistinguishable from the distribution of $\{P_0, P_1\}$ since the invertible affine transformations and the associated central map are randomly chosen at the time of running the key generation algorithm ME.KGen. Note that the distribution of $\{R'_0(\mathbf{x}'_0), R'_1(\mathbf{x}'_1)\}$ is indistinguishable from that of $\{R_0(\mathbf{x}_0), R_1(\mathbf{x}_1)\}$ under the hardness of MQ assumption. Hence the real view and the simulated view are indistinguishable. □

Theorem 4.3 *OT3 securely computes the functionality* $F_{\mathsf{OT3}} : (\{\mathbf{x}_0, \mathbf{x}_1\}, \sigma) \rightarrow (\perp, \mathbf{x}_\sigma)$ *against malicious sender and semi-honest receiver under the hardness of MQ assumption.*

Proof *Correctness:* Same as the correctness of OT1.

Consider the following two cases to prove the security of OT3.

Case I (Sender Sen *is corrupted):* Let the adversary $\mathcal{A}$ corrupt Sen in the real world and there be an incorruptible trusted third party (TTP) T in the ideal world apart from Sen and Rec. In order to simulate Rec in the ideal world, we design a simulator $\mathcal{SIM}$ that has oracle access to $\mathcal{A}$ and interacts with $\mathcal{A}$ as follows:

1. On receiving R from $\mathcal{A}$, the simulator $\mathcal{SIM}$ generates public key P by running the algorithm ME.KGen, sets $P_\sigma = P$, $P_{1-\sigma} = R - P_\sigma$ for its choice $\sigma \in \{0, 1\}$ and sends P_0 to $\mathcal{A}$.
2. The simulator $\mathcal{SIM}$, on receiving $P_0(\mathbf{x}_0), P_1(\mathbf{x}_1)$ and $\mathsf{IdP}\{\mathbf{x}_0, \mathbf{x}_1 | P_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge P_1(\mathbf{x}_1) = \mathbf{w}_1\}$ from $\mathcal{A}$, checks the correctness of the proof IdP. If the verification does not succeed then $\mathcal{SIM}$ outputs $\perp$ and aborts; otherwise, it runs the extractor algorithm to extract $\mathbf{x}_0, \mathbf{x}_1$ using a similar approach as given in Section 2.3.
3. The simulator $\mathcal{SIM}$ then behaves like an ideal world Sen by sending $\mathbf{x}_0, \mathbf{x}_1$ to T, who in turn computes the functionality $(\perp, x_\sigma)$ for the input $\sigma \in \{0, 1\}$ of ideal world Rec. Finally, $\mathcal{SIM}$ outputs whatever $\mathcal{A}$ outputs and terminates.

Table 1. Complexity of OT1.

	Sender	Receiver
Communication	$2n + \frac{n(n^2+3n+2)}{2}$ bit	$\frac{n(n^2+3n+2)}{2}$ bit
Computation	$n(n^2+3n+2)$ bit operations	$O(nd^2 \log d + d^3 + n^2)$ bit operations

Table 2. Complexity of OT2.

	Sender	Receiver
Communication	$2n + n(n^2+3n+2)$ bit	$\frac{n(n^2+3n+2)}{2}$ bit
Computation	$n(n^2+3n+2)$ bit operations	$O(nd^2 \log d + d^3 + n^2)$ bit operations

We first show that the honest receiver Rec's outputs in the real execution are indistinguishable from its outputs the ideal process. In the real execution, Rec outputs $\perp$ if verification of the proof IdP fails; otherwise, it outputs x_σ. Also in the ideal process, Rec outputs $\perp$ if the proof IdP is not valid; otherwise, it outputs x_σ. Thus, the distribution of the output of Rec is the same in the ideal process and in the real execution.

The adversary $\mathcal{A}$'s output contains its view and this view contains input $\{\mathbf{x}_0, \mathbf{x}_1\}$, the internal random coins and a set of multivariate polynomials P_0. The first two parts are indistinguishable in the real execution and in the ideal process. Moreover, the set of multivariate polynomials P_0 is indistinguishable in the two views since the invertible affine transformations (S,T) and the associated central map are randomly selected during running the key generation algorithm ME.KGen. Hence, we may conclude that the real view and the simulated view are indistinguishable.

Case II (Receiver Rec *is corrupted):* It is similar to Case II of security proof of OT1. □

Theorem 4.4 *OT4 securely computes the functionality* $F_{\mathsf{OT4}} : (\{\mathbf{x}_0, \mathbf{x}_1\}, \sigma) \rightarrow (\perp, \mathbf{x}_\sigma)$ *against malicious sender and semi-honest receiver under the hardness of the MQ assumption.*

Proof *Correctness:* Same as the correctness of OT2.

Similar to OT3, the security of OT4 can be proved by considering the following cases.

Case I (Sender Sen *is corrupted):* Let us assume that the adversary $\mathcal{A}$ corrupts Sen in the real world and T is an incorruptible TTP in the ideal world apart from Sen and Rec. Also, let the simulator $\mathcal{SIM}$ have the oracle access to $\mathcal{A}$. It then interacts with $\mathcal{A}$ to simulate Rec in the ideal world as follows.

1. The simulator $\mathcal{SIM}$, on receiving P_0, P_1 from $\mathcal{A}$, generates the algorithm ME.KGen to generate a public key R, computes $P_\sigma + R$ for its choice $\sigma \in \{0, 1\}$ and sends $P_\sigma + R$ to $\mathcal{A}$.
2. On receiving $R_0(\mathbf{x}_0)$, $R_1(\mathbf{x}_1)$ and $\mathsf{IdP}\{\mathbf{x}_0, \mathbf{x}_1 | R_0(\mathbf{x}_0) = \mathbf{w}_0 \bigwedge R_1(\mathbf{x}_1) = \mathbf{w}_1\}$ from $\mathcal{A}$, the simulator $\mathcal{SIM}$ checks the validity of the proof IdP. If the verification fails then $\mathcal{SIM}$ outputs $\perp$ and aborts, otherwise it runs the extractor algorithm to extract $\mathbf{x}_0, \mathbf{x}_1$ using a similar technique as given in Section 2.3.
3. In the following, the simulator $\mathcal{SIM}$ works as an ideal world Sen by sending $\mathbf{x}_0, \mathbf{x}_1$ to T, who in turn executes the functionality $(\perp, x_\sigma)$ for the input $\sigma \in \{0, 1\}$ of ideal world Rec. Finally, $\mathcal{SIM}$ outputs whatever $\mathcal{A}$ outputs and terminates.

Note that in both the ideal process and real execution, honest Rec outputs $\perp$ if the proof IdP is not valid and outputs x_σ if the proof IdP is correctly generated. Hence, Rec's outputs are indistinguishable in the ideal process and in the real execution.

On the other side, $\mathcal{A}$'s output contains its view. This view consists of input $\{\mathbf{x}_0, \mathbf{x}_1\}$, the internal random coins and a set of multivariate polynomials $P_\sigma + R$. The first two parts are indistinguishable in the two views. Since the invertible affine transformations (S,T) and the central map (F) are randomly selected during running the key generation algorithm ME.KGen, the set of multivariate polynomials $P_\sigma + R$ is indistinguishable in two views. Therefore, we can say that the simulated view and the real view are indistinguishable.

Case II (Receiver Rec *is corrupted):* It is similar to Case II of security proof of OT2. □

5. Efficiency

Communication complexity of our protocols is determined by counting the number of publicly transmitted bits, while the computation cost is evaluated by the MPKC encryptions and decryptions. Note that in each of our OT protocols, sender does two encryptions and receiver does one decryption. Tables 1 and 2 show the complexity of our OT1 and OT2, respectively, where the underlying encryption

scheme is considered as HFE [32] over binary field $\mathbb{F}_2$ and the degree of the HFE map over extension field $\mathbb{F}_{2^n}$ is considered as d.

6. Conclusion

In this work, we build two *post-quantum* OT protocols OT1 and OT2 depending on MPKC. Any MPKC-based encryption scheme can be employed as building blocks of our designs. Security of the proposed OT1 and OT2 is achieved in the semi-honest environment using random oracles. We then extended OT1 and OT2 to OT3 and OT4, respectively, by employing a 5-pass identification protocol of [31] in order to achieve security against the malicious sender. As far as we are aware of, our designs are the *first* MPKC-based OT protocols. Extending our OT protocols in the fully malicious model is an interesting direction of further research.

Appendix I. Security model

In order to design a cryptographic scheme, it is essential to prove its security. Informally, security is an attack game played between a challenger (simulator) and an attacker (adversary). Any multi-party protocol should satisfy the following basic security requirements:

Correctness: It ensures that an honest party receives the correct output at the end of the protocol.
Privacy: It indicates that on completion of the protocol, each party learns whatever is prescribed in the protocol, not beyond that.
Fairness: It ensures that a dishonest party learns its output only when the honest party learns its output.

Security model in semi-honest environment [35] A two-party protocol (Ω) is nothing but random process executing a function $\psi = (\psi_1, \psi_2) : (\chi, \zeta) \rightarrow (\psi_1(\chi, \zeta), \psi_2(\chi, \zeta))$. We say that the protocol Ω is secure in semi-honest environment if whatever can be evaluated by a party after involving in the protocol, which can be obtained using its input and output. In other words the parties follow the protocol honestly, while the adversaries try to extract additional information. It is formalized with the help of a simulation paradigm. During an execution of Ω on the input (χ, ζ), the view of a party P_i is denoted by $\mathsf{View}_i^{\Omega}(\chi, \zeta)$ and defined as $\mathsf{View}_i^{\Omega}(\chi, \zeta) = (\xi, r^{(i)}, \eta_1^{(i)}, ..., \eta_t^{(i)})$, where $\xi \in \{\chi, \zeta\}$ represents P_i's input, $r^{(i)}$ denotes the outcome of P_i's internal coin tosses and $\eta_l^{(i)}$ $(l = 1, 2, ..., t)$ stands for the l-th message obtained by P_i during the computation of Ω.

Definition Appendix I..1 Security in semi-honest Model:

Let the function $\psi = (\psi_1, \psi_2) : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^*$ be deterministic. Then we say that the protocol Ω computing ψ is secure in semi-honest environment if for PPT adversaries $\mathcal{SIM}_1$ (controlling P_1) and $\mathcal{SIM}_2$ (controlling P_2),

$$\{\mathcal{SIM}_1(\chi, \psi_1(\chi, \zeta))\}_{\chi,\zeta\in\{0,1\}^*} \stackrel{c}{\equiv} \mathsf{View}_1^{\Omega}(\chi, \zeta)_{\chi,\zeta\in\{0,1\}^*},$$
$$\{\mathcal{SIM}_2(\zeta, \psi_2(\chi, \zeta))\}_{\chi,\zeta\in\{0,1\}^*} \stackrel{c}{\equiv} \mathsf{View}_2^{\Omega}(\chi, \zeta)_{\chi,\zeta\in\{0,1\}^*},$$

where $\{\mathcal{SIM}_1(\chi, \psi_1(\chi, \zeta))\}_{\chi,\zeta\in\{0,1\}^*}$ and $\{\mathcal{SIM}_2(\zeta, \psi_2(\chi, \zeta))\}_{\chi,\zeta\in\{0,1\}^*}$, respectively, denote the simulated views of P_1 and P_2, which contain input of the corresponding party, simulated random coins and simulated protocol messages obtained by the corresponding party.

Security model in malicious environment [35]: In malicious model, adversaries can behave arbitrarily and can deviate at will from the prescribed protocol. In this case, security is formalized by an ideal process. An incorruptible trusted third party (TTP) involves in the ideal process. TTP gets the inputs from the participants, evaluates the functionality on those inputs and sends outputs back to them. We say that a protocol achieves its security in malicious model if a real world adversary can be simulated by an ideal world adversary. We discuss here the security framework of a two-party protocol in the presence of malicious adversary:

The real world: A protocol Ω is executed in the real world, where an honest party follows the instructions of Ω, while the adversary $\mathcal{A}_i$ (controlling P_i) can behave arbitrarily. Let P_1, P_2 have the private inputs χ, ζ and $\mathcal{A}_i$ have auxiliary input ϑ. On completion of the protocol Ω, an honest party outputs whatever is prescribed in Ω, a corrupted party outputs nothing and an adversary outputs the available transcripts as its view. In the real world, we denote the joint output as $\mathsf{REAL}_{\Omega,\mathcal{A}_i(\vartheta)}(\chi, \zeta)$.

The ideal process: Let the ideal process adversary $\mathcal{SIM}_i$ control P_i, where $i = 1, 2$. This process includes an incorruptible TTP.

Input: Suppose P_1, P_2 have inputs of χ, ζ, respectively, $\mathcal{SIM}_i$ has access to P_i's input and auxiliary input ϑ.
Sending inputs to TTP: An honest party always sends its real input to TTP, while a corrupted party may send arbitrary input or it can abort. Let TTP receive $(\overline{\chi}, \overline{\zeta})$, where $\overline{\chi}, \overline{\zeta}$ may not be equal to χ, ζ respectively. TTP sends $\perp$ to both the parties provided one of $\overline{\chi}, \overline{\zeta}$ is "abort".
TTP answers the adversary: TTP computes the functionality $\psi : (\overline{\chi}, \overline{\zeta}) \rightarrow (\psi_1(\overline{\chi}, \overline{\zeta}), \psi_2(\overline{\chi}, \overline{\zeta}))$ and sends $\psi_i(\overline{\chi}, \overline{\zeta})$ to $\mathcal{SIM}_i$, who in turn sends "abort" or "continue" to TTP.
TTP answers the honest party: If TTP receives "continue" from $\mathcal{SIM}_i$, then TTP sends $\psi_j(\overline{\chi}, \overline{\zeta})$ to the honest party

$(P_t, t \in \{1,2\} \setminus \{i\})$. Otherwise, TTP sends $\perp$ to $P_t, t \in \{1,2\} \setminus \{i\}$.
Output: The honest party $P_t, t \in \{1,2\} \setminus \{i\}$ always outputs the value whatever it receives from TTP, while the corrupted party P_i outputs nothing and the adversary outputs its view. In the ideal process, we denote the joint output as $\mathsf{IDEAL}_{\psi,\mathcal{SIM}_i(\vartheta)}(\chi,\zeta)$.

Definition Appendix I..2 *Simulatability:* Suppose Ω is a two-party protocol and ψ is the associated functionality. Then we say that Ω is secure in malicious environment if for any PPT real world adversary $\mathcal{A}_i$, there is a PPT ideal process adversary $\mathcal{SIM}_i$, such that $\mathsf{IDEAL}_{\psi,\mathcal{SIM}_i(Z)}(\chi,\zeta) \equiv^c \mathsf{REAL}_{\Omega,\mathcal{A}_i(\vartheta)}(\chi,\zeta)$ for each $i = 1, 2$.

References

[1] Rabin M O 2005 How to exchange secrets with oblivious transfer. *IACR Cryptology* e-print Archive p. 187

[2] Yao A C C 1986 How to generate and exchange secrets. In: *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (SFCS)*, IEEE, pp. 162–167

[3] Kushilevitz E and Ostrovsky R 1997 Replication is not needed: single database, computationally-private information retrieval. In: *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, IEEE, pp. 364–373

[4] Freedman M J, Ishai Y, Pinkas B and Reingold O 2005 Keyword search and oblivious pseudorandom functions. In: *Proceedings of the Theory of Cryptography Conference*. Berlin–Heidelberg: Springer, pp. 303–324

[5] Nielsen J B, Nordholt P S, Orlandi C and Burra S S 2012 A new approach to practical active-secure two-party computation. In: *Proceedings of the Annual Cryptology Conference*. Berlin: Springer, pp. 681–700

[6] Burra S S, Larraia E, Nielsen J B, Nordholt P S, Orlandi C, Orsini E, Scholl P and Smart N P 2015 High performance multi-party computation for binary circuits based on oblivious transfer. *IACR Cryptology* e-print Archive p. 472

[7] Shor P W 1999 Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review* 41(2): 303–332

[8] Garey M R and Johnson D S 1979 *Computers and intractability*. San Francisco: Freeman. 174

[9] Patarin J and Goubin L 1997 Trapdoor one-way permutations and multivariate polynomials. In: *Proceedings of the International Conference on Information and Communications Security*. Berlin: Springer, pp. 356–368

[10] Bogdanov A, Eisenbarth T, Rupp A and Wolf C 2008 Time-area optimized public-key engines: $\mathcal{MQ}$-cryptosystems as replacement for elliptic curves? In: *Proceedings of the International Workshop on Cryptographic Ware and Embedded Systems*. Berlin: Springer, pp. 45–61

[11] Chen A I T, Chen M S, Chen T R, Cheng C M, Ding J, Kuo E L H, Lee F Y S and Yang B Y 2009 SSE implementation of multivariate PKCs on modern x86 CPUs. In: *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems*. Berlin: Springer, pp. 33–48

[12] Dowsley R, Graaf J V D, Müller-Quade J and Nascimento A C 2008 Oblivious transfer based on the McEliece assumptions. In: *Proceedings of the International Conference on Information Theoretic Security*. Berlin: Springer, pp. 107–117

[13] McEliece R J 1978 A public-key cryptosystem based on algebraic. *Coding Thv* 4244: 114–116

[14] Kobara K, Morozov K and Overbeck R 2008 Coding-based oblivious transfer. In: *Mathematical Methods in Computer Science*. Berlin–Heidelberg: Springer, pp. 142–156

[15] David B M, Nascimento A C and Nogueira R B 2010 Oblivious transfer based on the McEliece assumptions with unconditional security for the sender. In: *Proceedings of X Simposio Brasileiro de Seguranca da Informac ao e de Sistemas Computacionais*

[16] Vasant S, Venkatesan S and Rangan C P 2012 A code-based 1-of-*N* oblivious transfer based on McEliece assumptions. In: *Proceedings of the International Conference on Information Security Practice and Experience*. Berlin: Springer, pp. 144–157

[17] David B M, Nascimento A C and Jr R T D S 2012 Efficient fully simulatable oblivious transfer from the McEliece assumptions. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 95(11): 2059–2066

[18] David B M, Nascimento A C and Müller-Quade J Universally composable oblivious transfer from lossy encryption and the McEliece assumptions. In: *Proceedings of the International Conference on Information Theoretic Security*. Berlin: Springer, pp. 80–99

[19] Peikert C, Vaikuntanathan V and Waters B 2008 A framework for efficient and composable oblivious transfer. In: *Proceedings of the Annual International Cryptology Conference* Berlin: Springer, pp. 554–571

[20] Lyubashevsky V, Palacio A and Segev G 2010 Public-key cryptographic primitives provably as secure as subset sum. In: *Proceedings of the Theory of Cryptography Conference*. Berlin: Springer, pp. 382–400

[21] Crépeau C and Kazmi R A 2015 Oblivious transfer from weakly random self-reducible public-key cryptosystem. In: *Proceedings of the International Symposium on Mathematical Foundations of Computer Science*. Berlin: Springer, pp. 261–273

[22] Zeng B, Tang X and Hsu C 2010 A framework for fully-simulatable *h*-of-*n* oblivious transfer. arXiv: 1005.0043

[23] Blazy O and Chevalier C 2015 Generic construction of UC-secure oblivious transfer. In: *Proceedings of the International Conference on Applied Cryptography and Network Security*. Cham: Springer, pp. 65–86

[24] Liu M and Hu Y 2019 Universally composable oblivious transfer from ideal lattice. *Frontiers of Computer Science* 13(4): 879–906

[25] Branco P, Ding J, Goulao M and Mateus P 2018 Universally composable oblivious transfer protocol based on the RLWE assumption. *IACR Cryptology* e-print Archive p. 1155

[26] Blazy O, Chevalier C and Vu Q H 2019 Post-quantum UC-secure oblivious transfer in the standard model with adaptive corruptions. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pp. 1–6

[27] Barreto P, Oliveira G and Benits W 2018 Supersingular isogeny oblivious transfer. arXiv preprint arXiv: 1805.06589

[28] Chou T and Orlandi C 2015 The simplest protocol for oblivious transfer. In: *Proceedings of the International Conference on Cryptology and Information Security in Latin America*. Cham: Springer, pp. 40–58

[29] Jao D and Feo L D 2011 Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: *Proceedings of the International Workshop on Post-Quantum Cryptography*. Berlin: Springer, pp. 19–34

[30] Branco P, Ding J, Goulao M and Mateus P 2019 A framework for universally composable oblivious transfer from one-round key-exchange. In: *Proceedings of the IMA International Conference on Cryptography and Coding*. Cham: Springer, pp. 78–101

[31] Sakumoto K, Shirai T and Hiwatari H 2011 Public-key identification schemes based on multivariate quadratic polynomials. In: *Proceedings of the Annual Cryptology Conference*. Berlin: Springer, pp. 706–723

[32] Patarin J 1996 Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In: *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*. Berlin: Springer, pp. 33–48

[33] Ding J, Gower J E and Schmidt D S 2006 *Multivariate public key cryptosystems*. Springer Science & Business Media, vol. 25

[34] Fiat A and Shamir A 1986 How to prove yourself: practical solutions to identification and signature problems. In: *Proceedings of the Conference on the Theory and Application of Cryptographic Techniques*. Berlin: Springer, pp. 186–194

[35] Goldreich O 2009 Basic applications. *Foundations of Cryptography*, vol. 2. Cambridge University Press