Efficient Oblivious Transfer from Lossy Threshold Homomorphic Encryption

Isheeta Nargis(✉)

University of Calgary, Calgary, Canada
inargis@ucalgary.ca

Abstract. In this article, a new oblivious transfer (OT) protocol, secure in the presence of erasure-free one-sided active adaptive adversaries is presented. The new bit OT protocol achieves better communication complexity than the existing bit OT protocol in this setting. The new bit OT protocol requires fewer number of public key encryption operations than the existing bit OT protocol in this setting. As a building block, a new two-party lossy threshold homomorphic public key cryptosystem is designed. It is secure in the same adversary model. It is of independent interest.

Keywords: Oblivious transfer · Active adversary · One-sided adaptive adversary · Threshold encryption · Lossy encryption · Public key encryption · Homomorphic encryption

1 Introduction

Oblivious transfer (OT) is one of the most critical problems in cryptography since many applications can be designed based on the existence of a secure OT protocol. In *one-sided active adaptive adversary model* for two-party computation, it is assumed that the adversary is active, adaptive and it can corrupt at most one party [13]. This is a relaxation from the standard adaptive adversary model for two-party computation, where the adversary can corrupt both parties. This relaxed model is used to achieve more efficient protocols. Let n denote the security parameter. Garay et al. [12] designed the most efficient OT protocol secure against active adaptive adversaries. For string OT of size q bits, their protocol requires $O(q)$ public key encryption (PKE) operations in the worst case. Here, q is a polynomial of n. Hazay and Patra [13] designed an OT protocol for one-sided active adaptive adversary model. For string OT of size q bits, their protocol requires constant number of PKE operations in the expected case. So, relaxing the notion of security has resulted in a protocol requiring significantly smaller number of PKE operations, in the expected case. In the *erasure-free adaptive adversary model*, it is assumed that the adversary can see all history of a party when it corrupts that party.

Hazay and Patra [13] designed an OT protocol for one-sided active adaptive adversary model. The OT protocol of [13] requires $O(n^2)$ communication

M. Joye and A. Nitaj (Eds.): AFRICACRYPT 2017, LNCS 10239, pp. 165–183, 2017.
DOI: 10.1007/978-3-319-57339-7_10

complexity for bit OT. One research goal is to improve the communication complexity in this setting.

Contribution of this Article. In this article, a new OT protocol secure against erasure-free one-sided active adaptive adversaries is presented. The worst case analysis is used as the measure of efficiency in this article. The new bit OT protocol needs $O(n)$ communication complexity, which is a significant improvement over the $O(n^2)$ communication complexity of the bit OT protocol of [13]. The bit OT protocol of [13] requires $O(n)$ PKE operations in the worst case, and the new bit OT protocol needs a constant number of PKE operations in the worst case. The OT protocol of [13] is secure in the universally composable (UC) framework. The new OT protocol is secure according to the simulation-based security definition of Canetti [5], which satisfies sequential composition.

As a building block, a new two-party lossy threshold homomorphic PKE scheme is designed. This encryption scheme is of independent interest. It can be used in other two-party protocols.

Techniques. Aumann and Lindell [1] designed an OT protocol secure against covert static adversaries. The new OT protocol is designed by converting their OT protocol. It is secure against erasure-free one-sided active adaptive adversaries. The new OT protocol achieves a much stronger notion of security than the OT protocol of [1] in two senses. Firstly, the active adversary model is a stronger security model than the covert adversary model [1]. Secondly, the adaptive adversary model is more secure than the static adversary model [5]. The OT protocol of [1] is modified in two ways. The protocol of [1] uses a traditional homomorphic PKE scheme and the new OT protocol uses a two-party lossy threshold homomorphic PKE scheme. For verification, the protocol of [1] uses cut-and-choose technique and the new OT protocol uses adaptive zero-knowledge arguments.

2 Background

Notation. Let n denote the security parameter. Let $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$ where q is a prime. Let $\mathbb{Z}_q^* = \{1, 2, \ldots, q-1\}$. For all elements a and $b \neq 1$ in group $\mathbb{G}$, the discrete logarithm of a in base b is denoted by $\log_b(a)$. For a set R, let $r \xleftarrow{\$} R$ denote that r is selected uniformly at random from R. Let A be a probabilistic polynomial-time algorithm. Let $coins(A)$ denote the distribution of the internal randomness of A. $y \leftarrow A(x)$ means that y is computed by running A on input x and randomness r where $r \xleftarrow{\$} coins(A)$. Let $E_{pk}(m, r)$ denote the result of encryption of plaintext m using encryption key pk and randomness r. Let $D_{sk}(c)$ denote the result of decryption of ciphertext c using decryption key sk. Let $Com_{\mu}(a, r)$ denote the commitment of secret a using commitment key μ and randomness r.

The DDH Assumption. The decisional Diffie-Hellmann (DDH) assumption for cyclic group $\mathbb{G}$ of order prime q says that, for random generator $g \in \mathbb{G}^*$

($\mathbb{G}^*$ denotes the generators of $\mathbb{G}$), the tuples (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^c) are computationally indistinguishable, where $a, b, c \xleftarrow{\$} \mathbb{Z}_q$.

Trapdoor Commitment Scheme. A *trapdoor commitment scheme* is a commitment scheme such that a trapdoor is generated during the key generation. With the trapdoor, one can efficiently compute a randomness to open a given commitment to any value of choice. Without the trapdoor, the binding property of the commitment scheme holds. Pedersen [18] designed a trapdoor commitment scheme based on the DDH assumption. In Pedersen's commitment scheme, the commitment key is $\mu = g^{\delta}$, and δ is the trapdoor.

Adaptive Zero-Knowledge Arguments. For definition of zero-knowledge argument, see [15]. An adaptive zero-knowledge argument is a zero-knowledge argument secure against adaptive adversaries. For definition of non-erasure Σ-protocol, see [8,17].

Additive Homomorphic PKE Scheme. In an *additive homomorphic PKE scheme*, one can efficiently compute an encryption c of $(m_1 + m_2)$ from ciphertexts c_1 and c_2 encrypting plaintexts m_1 and m_2, respectively. This is called *homomorphic addition* and denoted by $c = c_1 +_h m_2$. In an additive homomorphic PKE scheme, one can also efficiently compute an encryption c_2 of $(m_1 \times m_2)$ from an encryption c_1 of m_1 and the plaintext m_2. This is called *homomorphic multiplication by constant*, and denoted by $c_2 = m_2 \times_h c_1$.

Randomizable PKE Scheme. In a *randomizable PKE scheme*, there exists a probabilistic polynomial-time algorithm *Blind*, which, on input public key pk and an encryption c of plaintext m, produces another encryption c_1 of plaintext m such that c_1 is distributed identically to $E_{pk}(m, r)$ where $r \xleftarrow{\$} Coins(E)$.

Lossy Encryption Scheme

Definition 1 *(Lossy PKE Scheme [3]). A* ***lossy PKE scheme*** *is a tuple (G, E, D) of probabilistic polynomial time algorithms such that keys generated by $G(1^k, 1)$ and $G(1^k, 0)$ are called injective keys and lossy keys, respectively. The algorithms must satisfy the following properties.*

1. ***Correctness on Injective Keys:*** *For all plaintexts m,*

$$Pr\Big[(pk, sk) \leftarrow G(1^k, 1); r \xleftarrow{\$} coins(E) : D_{sk}\big(E_{pk}(m, r)\big) = m\Big] = 1.$$

2. ***Indistinguishability of Keys:*** *The lossy public keys are computationally indistinguishable from the injective public keys. If $proj : (pk, sk) \rightarrow pk$ is the projection map, then $\{proj(KG(1^k, 1))\} \stackrel{c}{\equiv} \{proj(KG(1^k, 0))\}$.*
3. ***Lossiness on Lossy Keys:*** *If $(pk_\ell, sk_\ell) \leftarrow G(1^k, 0)$, then, for all m_0, m_1, the distributions $E_{pk_\ell}(m_0, R)$ and $E_{pk_\ell}(m_1, R)$ are statistically indistinguishable.*

4. ***Openability:*** *If* $(pk_\ell, sk_\ell) \leftarrow G(1^k, 0)$ *and* $r_0 \overset{\$}{\leftarrow} coins(E)$*, then, for all* m_0, m_1*, with overwhelming probability, there exists* $r_1 \in coins(E)$ *such that* $E_{pk_\ell}(m_0, r_0) = E_{pk_\ell}(m_1, r_1)$*. That is, there exists a (possibly inefficient) algorithm Opener that can open a lossy ciphertext to any arbitrary plaintext with all but negligible probability.*

The semantic security of a lossy encryption scheme is implied by definition [3].

Security Model. The security of the new protocols are proved following the simulation based security definition by Canetti [5].

3 Definition of Two-Party Lossy Threshold PKE Scheme

A definition of two-party lossy threshold PKE scheme secure against one-sided active adaptive adversaries is presented below.

Definition 2 *(Lossy Threshold PKE Scheme Secure against Erasure-Free One-Sided Active Adaptive Adversaries). A* ***lossy threshold PKE scheme secure against erasure-free one-sided active adaptive adversaries*** *for the set of parties* $P = \{P_1, P_2\}$*, and security parameter* n*, is a 4-tuple* (K, KG, E, Π_{DEC}) *having the following properties.*

Key Space: *The key space* K *is a family of finite sets* (pk, sk_1, sk_2)*.* pk *is the public key and* sk_i *is the secret key share of* P_i*. Let* M_{pk} *denote the message space for public key* pk*.*

Key Generation: *There exists a probabilistic polynomial-time key generation algorithm* KG*, which, on input* $(1^n, mode)$*, generates public output* pk *and a list* $\{vk, vk_1, vk_2\}$ *of verification keys, and secret output* sk_i *for* P_i*, where* $(pk, sk_1, sk_2) \in K$*. By setting mode to zero and one, key in lossy mode and injective mode can be generated, respectively.* vk *is called the verification key,* vk_i *is called the verification key of* P_i*.*

Encryption: *There exists a probabilistic polynomial-time encryption algorithm* E*, which, on input* pk*,* $m \in M_{pk}$*,* $r \overset{\$}{\leftarrow} coins(E)$*, outputs an encryption* $c = E_{pk}(m, r)$ *of* m*.*

Decryption: *There exists a two-party decryption protocol* Π_{DEC} *secure against erasure-free one-sided active adaptive adversaries. On common public input* (c, pk, vk, vk_1, vk_2)*, and secret input* sk_i *for each* $P_i, i \in \{1, 2\}$*, where* sk_i *is the secret key share of* P_i *for the public key* pk *(as generated by* KG*), and* c *is an encrypted message,* Π_{DEC} *returns a message* m*, or the symbol* $\perp$ *denoting a decryption failure, as a common public output.*

Lossy Encryption Properties: *The encryption scheme is a lossy encryption scheme according to Definition 1.*

Threshold Semantic Security: *Consider the following game* G *for an erasure-free one-sided active adaptive adversary* $\mathcal{A}$*.*

G1. *$\mathcal{A}$ may corrupt at most one party. If $\mathcal{A}$ corrupts P_i, then $\mathcal{A}$ learns the history of P_i.*

G2. *The challenger executes algorithm KG. The challenger broadcasts the public key and the verification keys. For each $i \in \{1, 2\}$, the challenger sends sk_i to P_i. If there is a corrupted party P_i, then $\mathcal{A}$ learns sk_i.*

G3. *$\mathcal{A}$ adaptively makes the following types of queries.*

1. *Corruption query*
   $\mathcal{A}$ may corrupt a party, if no party was corrupted before. If $\mathcal{A}$ corrupts P_i, then $\mathcal{A}$ learns sk_i and the history of P_i.
2. *Decryption query*
   $\mathcal{A}$ selects a message $m \in M_{pk}$, and sends it to the challenger. The challenger sends $\mathcal{A}$ the decryption shares and the validity proofs of P_1 and P_2, for an encryption of m.
   $\mathcal{A}$ repeats step G3 as many times as it wishes.

G4. *$\mathcal{A}$ selects two message m_0 and m_1 from M_{pk}, and sends them to the challenger. The challenger randomly selects a bit b, and sends an encryption c of m_b, to $\mathcal{A}$.*

G5. *$\mathcal{A}$ repeats step G3 as many times as it wishes. $\mathcal{A}$ cannot request message m_0 or m_1 in step G3(2).*

G6. *$\mathcal{A}$ outputs a guess bit b_1.*

A threshold encryption scheme is said to be ***semantically secure against erasure-free one-sided active adaptive adversaries*** *if, for any probabilistic polynomial-time erasure-free one-sided active adaptive adversary, $b = b_1$ with probability only negligibly greater than $\frac{1}{2}$.*

The verification keys are used for validity proofs in Π_{DEC}. During Π_{DEC}, each party P_i uses *validity proof* such that P_i can convince the remaining party that P_i performed its calculation in Π_{DEC} correctly, without disclosing its secret. Note that $\mathcal{A}$ can only request for ciphertexts for which it knows the plaintext. It is not like the chosen ciphertext attack (CCA) where the adversary can ask for decryption shares for any chosen ciphertext. Step G3(2) is used in game G to denote that, despite learning all the decryption shares and validity proofs for several chosen plaintexts, the adversary still does not gain any advantage in guessing the plaintext from the ciphertext.

Let $\mathcal{F}_{KG}$ be the ideal functionality for the key generation. In a two-party lossy threshold encryption scheme, there may exist a two-party distributed key generation (DKG) protocol that computes $\mathcal{F}_{KG}$ securely against erasure-free one-sided active adaptive adversaries.

4 A New Two-Party Lossy Threshold Homomorphic Encryption Scheme

In this section, a new two-party lossy threshold homomorphic public key encryption scheme $ELTA2E = (K, KG, E, \Pi_{DEC})$ is presented. The name $ELTA2E$ denotes an encryption scheme that is lossy, threshold, secure against adaptive adversaries, for two parties and based on the ElGamal encryption scheme.

$ELTA2E$ is based on the DDH assumption. All protocols of $ELTA2E$ work in the CRS model.[1]

Bellare and Yilek [4] designed a non-threshold lossy encryption scheme with efficient *Opener* algorithm, based on the DDH assumption. Let *EncLossy* denote their encryption scheme. $ELTA2E$ is created by adding the threshold properties to *EncLossy*.

One possible group $\mathbb{G}$ for $ELTA2E$ is as follows. *Safe primes* are primes of the form $p = 2q + 1$ where q is also a prime. On input n, using known methods to generate safe primes, an n-bit safe prime p is generated, with generator g_0 of $\mathbb{Z}_p^*$. There is exactly one subgroup $\mathbb{G}$ of $\mathbb{Z}_p^*$ of order q. Let g be the generator of $\mathbb{G}$. $g = g_0^{\frac{p-1}{q}} = (g_0)^2$. (p, q, g) is the description of group $\mathbb{G}$. Unless otherwise specified, all computations are performed in group $\mathbb{G}$. Pedersen commitment scheme [18] is used as the trapdoor commitment scheme in $ELTA2E$.

Key Generation. Let the input be $(1^n, mode)$. Select $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_q$. Set $\alpha = (\alpha_1 + \alpha_2) \bmod q, h = g^\alpha, h_1 = g^{\alpha_1}, h_2 = g^{\alpha_2}$. Select $\gamma \xleftarrow{\$} \mathbb{Z}_q$. Set $j = g^\gamma$. If $mode = 1$, then set $\ell = g^{\gamma\alpha}$. If $mode = 0$, then select $\rho \xleftarrow{\$} \mathbb{Z}_q \setminus \{\alpha\}$, and set $\ell = g^{\gamma\rho}$. The public key is $pk = (q, g, j, h, \ell)$. The secret key shares are $sk_1 = \alpha_1, sk_2 = \alpha_2$. The verification keys are $vk = g, vk_1 = h_1, vk_2 = h_2$.

Encryption. The encryption algorithm E works as follows. Let the plaintext be $m \in \{0, 1\}$. Select randomness $r = (s, t) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$. Compute $y = g^s j^t$, and $z = h^s \ell^t g^m$. Return the ciphertext $c = (y, z)$.

Protocol for Threshold Decryption. The threshold decryption protocol Π_{DEC} works as follows. P_1 sends $ds_1 = y^{(sk_1)}$. Adaptive zero-knowledge argument for equality of discrete logarithm is used as the validity proof in Π_{DEC}. P_1 proves that $\log_y(ds_1) = \log_{vk}(vk_1)$. If P_1 fails, then P_2 aborts. P_2 sends $ds_2 = y^{(sk_2)}$. P_2 proves that $\log_y(ds_2) = \log_{vk}(vk_2)$. If P_2 fails, then P_1 aborts. P_1 and P_2 compute $w = \frac{z}{ds_1 \cdot ds_2}$. From w, P_1 and P_2 compute m where $m \in \{0, 1\}$, and $g^m = w$ in $\mathbb{G}$. If there is no such value m, then P_1 and P_2 output $\perp$, denoting decryption failure.

It is also possible to perform *private threshold decryption* to just one party P_k. In that case, P_{2-k} sends $ds_{2-k} = y^{(sk_{2-k})}$, and proves as above. P_k computes ds_k, then computes the output similarly.

Distributed Key Generation Protocol. $ELTA2E$ has a DKG protocol Π_{DKG}. The protocol Π_{DKG} is presented below.

Protocol Π_{DKG}.

CRS: $\mu \in \mathbb{Z}_p$.
Group description: (p, q, g).
Input: $(1^n, mode)$.

[1] In the common reference string (CRS) model, it is assumed that all parties have access to a common string that is selected from some specified distribution.

1. P_1 selects $\alpha_1, \gamma_1, \beta_1, \theta_1 \xleftarrow{\$} \mathbb{Z}_q$. P_1 sets $sk_1 = \alpha_1$. P_1 computes $h_1 = g^{\alpha_1}, j_1 = g^{\gamma_1}$. P_1 computes commitments $b_1 = Com_\mu(h_1, \beta_1), c_1 = Com_\mu(j_1, \theta_1)$. P_1 sends (b_1, c_1).
2. P_1 proves the knowledge of committed secret for commitments b_1 and c_1. If P_1 fails in any proof, then P_2 aborts.
3. P_2 selects $\alpha_2, \gamma_2 \xleftarrow{\$} \mathbb{Z}_q$. P_2 sets $sk_2 = \alpha_2$. P_2 computes $h_2 = g^{\alpha_2}, j_2 = g^{\gamma_2}$. P_2 sends (h_2, j_2).
4. P_2 proves knowledge of discrete logarithm of h_2 and j_2. If P_2 fails in any proof, then P_1 aborts.
5. P_1 sends the openings (h_1, β_1) and (j_1, θ_1) of its commitments.
6. P_2 verifies that $b_1 = Com_\mu(h_1, \beta_1)$, and $c_1 = Com_\mu(j_1, \theta_1)$. If any of these two equalities does not hold, then P_2 aborts.
7. P_1 and P_2 set $vk = g, vk_1 = h_1, vk_2 = h_2, h = h_1 h_2, j = j_1 j_2$.
8. If $mode = 0$, then P_1 selects $\tau_1 \xleftarrow{\$} \mathbb{Z}_q \setminus \{\alpha_1\}$, and sets $\ell_1 = j^{\tau_1}$.
   If $mode = 1$, then P_1 sets $\ell_1 = j^{\alpha_1}$. P_1 sends ℓ_1.
9. If $mode = 1$, then P_1 proves that $\log_j(\ell_1) \neq \log_{vk}(vk_1)$.
   If $mode = 1$, then P_1 proves that $\log_j(\ell_1) = \log_{vk}(vk_1)$.
   If P_1 fails, then P_2 aborts.
10. P_2 sends $\ell_2 = j^{\alpha_2}$.
11. P_2 proves that $\log_j(\ell_2) = \log_{vk}(vk_2)$. If P_2 fails, then P_1 aborts.
12. P_1 and P_2 set $\ell = \ell_1 \ell_2, pk = (q, g, j, h, \ell)$.
13. P_1 outputs $(pk, sk_1, (vk, vk_1, vk_2))$.
14. P_2 outputs $(pk, sk_2, (vk, vk_1, vk_2))$.

The proofs in steps 2,4,9, and 11 are performed using adaptive zero-knowledge arguments. The CRS μ is used as the commitment key for Pedersen commitment scheme. The CRS μ also acts as the CRS for the zero-knowledge arguments. The reason for using commitments in Π_{DKG} is to ensure that no party can affect the distribution of the generated key.

Lemma 1. *ELTA2E is additive homomorphic.*

Proof. **Homomorphic Addition.** Let $c_1 = (g^{s_1} j^{t_1}, h^{s_1} \ell^{t_1} g^{m_1})$, and $c_2 = (g^{s_2} j^{t_2}, h^{s_2} \ell^{t_2} g^{m_2})$ be two ciphertexts encrypting plaintexts m_1 and m_2, respectively. $c = c_1 +_h c_2 = (g^{s_1} j^{t_1} \cdot g^{s_2} j^{t_2}, h^{s_1} \ell^{t_1} g^{m_1} \cdot h^{s_2} \ell^{t_2} g^{m_2}) = (g^{s_1+s_2} j^{t_1+t_2}, h^{s_1+s_2} \ell^{t_1+t_2} g^{m_1+m_2})$.

Homomorphic Multiplication by Constant. Let $c_1 = (y_1, z_1) = (g^{s_1} j^{t_1}, h^{s_1} \ell^{t_1} g^{m_1})$ be a ciphertext encrypting plaintext m_1. Let m_2 be a known plaintext. $c_2 = c_1 \times_h m_2 = \left((g^{s_1} j^{t_1})^{m_2}, (h^{s_1} \ell^{t_1} g^{m_1})^{m_2}\right) = (g^{s_1 m_2} j^{t_1 m_2}, h^{s_1 m_2} \ell^{t_1 m_2} g^{m_1 m_2})$.

ELTA2E **is Randomizable.** Let $c = (y, z) = (g^s j^t, h^s \ell^t g^m)$ be a ciphertext encrypting plaintext m. The *Blind* function on input $(pk, c) = ((q, g, j, h, -\ell), (y, z))$ works as follows. Select $s_1, t_1 \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$. Set $y_1 = y \cdot g^{s_1} j^{t_1}, z_1 = z \cdot h^{s_1} \ell^{t_1}$. Return $c_1 = (y_1, z_1)$.

5 Security of the DKG Protocol Π_{DKG}

Canetti et al. [6] introduced the *single inconsistent party* (SIP) technique. At the start of simulation, the identity of the single inconsistent party (SIP) is generated uniformly at random. The view of any party except the SIP in the simulation is computationally indistinguishable from its view in the real world. The view of the adversary is independent from the choice of SIP. This technique is used in the security proofs of $ELTA2E$. Let $\mathcal{A}$ be a one-sided active adaptive adversary and $\mathcal{Z}$ be the environment. Let $\mathcal{S}_{DKG}$ be the simulator for Π_{DKG} for adversary $\mathcal{A}$ and environment $\mathcal{Z}$. At start, $\mathcal{S}_{DKG}$ selects $I \xleftarrow{\$} \{P_1, P_2\}$ where I denotes the identity of the SIP. If $\mathcal{A}$ corrupts I, then $\mathcal{S}_{DKG}$ rewinds to the start of simulation, generates a new $I \xleftarrow{\$} \{P_1, P_2\}$, and proceeds again. $\mathcal{A}$ corrupts at most one party, so the probability of a randomly selected party I being corrupted is at most $\frac{1}{2}$. The expected number of rewinds of $\mathcal{S}_{DKG}$ is at most two, and the simulation can be performed in expected polynomial time. To bound the running time of $\mathcal{S}_{DKG}$ to strictly polynomial time, simulation can continue running up to n^{ℓ_1} steps where ℓ_1 is a constant. If $\mathcal{S}_{DKG}$ does not halt within n^{ℓ_1} steps, then $\mathcal{S}_{DKG}$ fails. The probability of failure of $\mathcal{S}_{DKG}$ is negligible.

Theorem 1. *Provided that the DDH assumption holds, and trapdoor commitment scheme and adaptive zero-knowledge arguments exist, protocol Π_{DKG} computes $\mathcal{F}_{KG}$ securely against erasure-free one-sided active adaptive adversaries.*

Proof (Sketch). The main idea of the proof is given here. The full proof is available in the full version [16]. At start, $\mathcal{S}_{DKG}$ selects $\delta \xleftarrow{\$} \mathbb{Z}_q$, and sets the CRS to $\mu = g^{\delta}$. Then, $\mathcal{S}_{DKG}$ knows the trapdoor δ of the commitment key μ. The simulator $\mathcal{S}_{DKG}$ for the case where P_1 is the SIP, works as follows. If $\mathcal{A}$ corrupts P_2 after any step, then $\mathcal{S}_{DKG}$ corrupts P_2 in the ideal world. If P_2 fails in some proof, then P_1 aborts in the real world. In the ideal world, $\mathcal{S}_{DKG}$ sends $abort_{P_2}$ to the trusted party and halts. The trusted party sends $abort_{P_2}$ to P_1, and P_1 halts. If P_2 does not fail in any proof, then the following things happen. In step 1, $\mathcal{S}_{DKG}$ selects $\alpha_2, \gamma_2 \xleftarrow{\$} \mathbb{Z}_q$, computes $h_2 = g^{\alpha_1}, j_2 = g^{\gamma_2}, h_1 = \frac{h}{h_2}, j_1 = \frac{j}{j_2}$. $\mathcal{S}_{DKG}$ selects $\beta_1, \theta_1 \xleftarrow{\$} \mathbb{Z}_q$ and computes $b_1 = Com_{\mu}(h_1, \beta_1), c_1 = Com_{\mu}(j_1, \theta_1)$. By the hiding property of the commitment scheme, the distribution of (b_1, c_1) in two worlds are identical. $\mathcal{S}_{DKG}$ honestly performs step 2. In step 3, if P_2 is honest, then $\mathcal{S}_{DKG}$ uses h_2, j_2 computed in step 1. In step 4 and 11, if P_2 is corrupted, $\mathcal{S}_{DKG}$ acts as an honest verifier. If P_2 passes the proofs, then $\mathcal{S}_{DKG}$ extracts α_2 and γ_2 using the knowledge extractor of the zero-knowledge argument, in step 4. If P_2 is honest, then $\mathcal{S}_{DKG}$ acts as an honest prover in step 4 and 11. In step 5, $\mathcal{S}_{DKG}$ computes $\hat{h}_1 = \frac{h}{h_2}, \hat{j}_1 = \frac{j}{j_2}$. Using the trapdoor δ of the commitment key μ, $\mathcal{S}_{DKG}$ computes $\hat{\beta}_1, \hat{\theta}_1$ such that $b_1 = Com_{\mu}(\hat{h}_1, \hat{\beta}_1)$, and $c_1 = Com_{\mu}(\hat{j}_1, \hat{\theta}_1)$. $\mathcal{S}_{DKG}$ uses $(\hat{h}_1, \hat{\beta}_1), (\hat{j}_1, \hat{\theta}_1)$ as the message from P_1. If $\mathcal{A}$ corrupts P_2 before step 3, then, corrupted P_2 sends h_2 and j_2 in step 3. The value of h and j are fixed since they are part of the input of $\mathcal{S}_{DKG}$. $\mathcal{A}$ sees that the openings of the commitments are consistent, and $h = \hat{h}_1 h_2$ and $j = \hat{j}_1 j_2$, as required. $(\hat{h}_1, \hat{j}_1)$ is identically distributed to (h_1, j_1). If P_2 is

honest up to step 3, then $\hat{h}_1 = \frac{h}{h_2} = h_1, \hat{\beta}_1 = \beta_1, \hat{j}_1 = j_1, \hat{\theta}_1 = \theta_1$. In step 6, P_1 passes the verification tests in the ideal world. In step 7, $\mathcal{S}_{DKG}$ sets $vk = g, vk_1 = \hat{h}_1, vk_2 = h_2$, and uses (h, j) of the input. As argued earlier, these values in two worlds are identically distributed. In step 8, $\mathcal{S}_{DKG}$ computes $\hat{\ell}_1 = \frac{\ell}{j^{(\alpha_2)}} > \ell$ is fixed in both worlds. The distribution of α_2 in two worlds are identical. Then, $\hat{\ell}_1$ and ℓ_1 are identically distributed. In step 9, $\mathcal{S}_{DKG}$ generates a proof transcript using trapdoor δ. By definition of zero-knowledge argument, the proof transcript in two worlds are computationally indistinguishable. If P_2 is honest, then $\mathcal{S}_{DKG}$ honestly performs steps 10 and 11. In step 12, $\mathcal{S}_{DKG}$ uses (ℓ, pk) of input. In step 13, the output of honest P_1 is $(pk, vk, vk_1, vk_2, \alpha_1)$. Then, the output of the honest P_1 in two worlds are identically distributed. In step 14, if P_2 is honest, then $\mathcal{S}_{DKG}$ sets $(pk, vk, vk_1, vk_2, \alpha_2)$ as the output of P_2. The distribution of sk_2 in two worlds are identical. The simulator for the case where P_2 is the SIP is similar, so it is not given separately.

6 Security of Encryption Scheme *ELTA2E*

Lemma 2. *If the decisional Diffie-Hellman assumption holds, then ELTA2E is a lossy encryption scheme. ELTA2E has an efficient (polynomial-time) Opener algorithm.*

Proof. **Correctness of Decryption in the Injective Mode.** In the injective mode, $pk = (q, g, j, h, \ell) = (q, g, g^\gamma, g^\alpha, g^{\alpha \cdot \gamma})$. Then, $w = \frac{z}{ds_1 \cdot ds_2} = \frac{z}{y^{sk_1} \cdot y^{sk_2}} = \frac{z}{y^{\alpha_1 + \alpha_2}} = \frac{z}{y^\alpha} = \frac{h^s \ell^t g^m}{(g^s j^t)^\alpha} = \frac{(g^\alpha)^s (g^{\gamma\alpha})^t g^m}{\left(g^s (g^\gamma)^t\right)^\alpha} = \frac{g^{\alpha s + \alpha\gamma t + m}}{g^{\alpha s + \alpha\gamma t}} = g^m$.

Indistinguishability of Keys. In the injective mode, $pk = (q, g, j, h, \ell) = (q, g, g^\gamma, g^\alpha, g^{\gamma\alpha})$. In the lossy mode, $pk = (q, g, j, h, \ell) = (q, g, g^\gamma, g^\alpha, g^{\gamma\rho})$. By the DDH assumption, the public key in injective mode is computationally indistinguishable from the public key in lossy mode.

Lossiness on Lossy Keys. Let $pk = (q, g, j, h, \ell) = (q, g, g^\gamma, g^\alpha, g^{\gamma\rho})$ be a lossy public key. Encryption of a message m with randomness (s, t) is $c = (y, z) = (g^{s+\gamma t}, g^{\alpha s + \gamma\rho t} \cdot g^m)$. Since $\rho \neq \alpha$, $(s+\gamma t)$ and $(\alpha s + \gamma\rho t)$ are linearly independent combinations of s and t. Then, y and z are uniformly random group elements.

Efficient *Opener* **Algorithm.** Let $pk = (q, g, j, h, \ell) = (q, g, g^\gamma, g^\alpha, g^{\gamma\rho})$ be a lossy public key. Let the corresponding secret key be $sk = (\gamma, \rho, \alpha)$. Let $c = (y, z)$ be an encryption of plaintext m with randomness $r = (s, t)$. Then, $c = (y, z) = (g^{s+\gamma t}, g^{\alpha s + \gamma\rho t} \cdot g^m)$. Let m_1 be the plaintext with which the ciphertext c has to be opened. On input $(pk, sk, (y, z), m, (s, t), m_1)$, the algorithm *Opener* has to find randomness $r_1 = (s_1, t_1) \in \mathbb{Z}_q \times \mathbb{Z}_q$ such that $s + \gamma t = s_1 + \gamma t_1$, and $\alpha s + \gamma\rho t + m = \alpha s_1 + \gamma\rho t_1 + m_1$. These two equations are two linear equations on the variables (s_1, t_1). The *Opener* algorithm solves these two equations to find s_1 and t_1 in polynomial time.

Lemma 3. *Provided that the decisional Diffie-Hellman assumption holds, and trapdoor commitment scheme and adaptive zero-knowledge arguments exist, the encryption scheme ELTA2E achieves threshold semantic security in the presence of erasure-free one-sided active adaptive adversaries.*

Proof (Sketch). The threshold semantic security is proved by reduction, following the idea in [11]. The lossy encryption properties of *EncLossy* are proved in [4]. Since any lossy PKE scheme is semantically secure [3], *EncLossy* is semantically secure. Assume that there exists a probabilistic polynomial-time one-sided active adaptive adversary $\mathcal{A}_1$ that can break the semantic security of the two-party lossy threshold encryption scheme *ELTA2E*. It is described how to construct a probabilistic polynomial-time one-sided active adaptive adversary $\mathcal{A}_2$, using $\mathcal{A}_1$, that can break the semantic security of the non-threshold lossy encryption scheme *EncLossy*. As *EncLossy* is semantically secure, a contradiction is reached. To convert $\mathcal{A}_1$ to $\mathcal{A}_2$, it is necessary to simulate the extra information that are not available in the non-threshold lossy cryptosystem. The simulator is designed using the SIP technique. The inputs of the simulator are the public key $pk = (q, g, j, h, \ell)$, the mode parameter *mode*, and the identity I of the single inconsistent party. In step G1, if $\mathcal{A}_1$ corrupts a party P_i, then $\mathcal{A}_2$ corrupts P_i. $\mathcal{A}_2$ receives the history of P_i from $\mathcal{Z}$. In step G2, if P_1 is the SIP, $\mathcal{A}_2$ simulates as follows. $\mathcal{A}_2$ selects $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_q$. $\mathcal{A}_2$ sets $sk_1 = \alpha_1, sk_2 = \alpha_2, vk = g, vk_2 = g^{\alpha_2}, vk_1 = \frac{h}{vk_2}$. $\mathcal{A}_2$ sends $((pk, vk, vk_1, vk_2, sk_1), (pk, vk, vk_1, vk_2, sk_2))$ to $\mathcal{A}_1$ in step G2. The distribution of sk_1, sk_2 are identical in two worlds. $\mathcal{A}_2$ sets $vk_1 = \frac{h}{vk_2}$. The value of h is fixed and the distribution of vk_2 in two worlds are identical. Therefore, the distribution of vk_1 in two worlds are identical. Here $h = vk_1 \cdot vk_2$, so it is consistent. If the adversary corrupts P_2, then it sees that $vk_2 = g^{sk_2}$ so everything is consistent for P_2. When P_2 is the SIP, the simulation is similar and not given separately. In step G3(1), if $\mathcal{A}_1$ corrupts a party P_i, then $\mathcal{A}_2$ corrupts P_i. $\mathcal{A}_2$ receives sk_i and the history of P_i from $\mathcal{Z}$. In step G3(2), $\mathcal{A}_1$ selects $m \in M_{pk}$ and sends m to $\mathcal{A}_2$. $\mathcal{A}_2$ computes $c_m = (y_m, z_m) = (g^s j^t, h^s \ell^t g^m)$. c_m is a valid encryption of m. If P_1 is the SIP, then $\mathcal{A}_2$ simulates the steps of protocol Π_{DEC} as follows. In step 1, $\mathcal{A}_2$ computes $ds_1 = (y_m)^{sk_1}$ where sk_1 is the secret key share of P_1 computed by $\mathcal{A}_2$ in step G2. As argued in step G2, the distribution of sk_1 in two worlds are identical. Then, the distribution of ds_1 in two worlds are identical. In step 2, $\mathcal{A}_2$ acts as an honest prover. In step 3, if P_2 is honest, then $\mathcal{A}_2$ computes $ds_2 = (y_m)^{sk_2}$ where sk_2 is the secret key share of P_2 computed by $\mathcal{A}_2$ in step G2. Proof argument is similar to step 1. In step 4, if P_2 is honest, then $\mathcal{A}_2$ acts as an honest prover. If P_2 is corrupted, then $\mathcal{A}_2$ acts as an honest verifier. If P_2 fails, then $\mathcal{A}_2$ sends $abort_{P_2}$ to the trusted party and halt. Then, the trusted party sends $abort_{P_2}$ to P_1 and honest P_1 halts. Honest P_1 aborts in the real world. In step 5, $\mathcal{A}_2$ computes $w = g^m$. The value of w is identical in two worlds. In step 6, $\mathcal{A}_2$ uses m. The simulation of step G3(2) when P_2 is the SIP is similar. So, it is not given separately. In step G4, $\mathcal{A}_1$ chooses two plaintexts $m_0, m_1 \in M_{pk}$ and sends them to $\mathcal{A}_2$. $\mathcal{A}_2$ sends (m_0, m_1) to the challenger of *EncLossy*. Then, the challenger of *EncLossy* selects $b \xleftarrow{\$} \{0, 1\}$, computes an

encryption c of m_b and returns c to $\mathcal{A}_2$. $\mathcal{A}_2$ sends c to $\mathcal{A}_1$. Step G5 is similar to step G3. In step G6, $\mathcal{A}_1$ returns a guess b_1. $\mathcal{A}_2$ returns b_1.

Theorem 2. *Provided that the DDH assumption holds, and trapdoor commitment scheme and adaptive zero-knowledge arguments exist, the encryption scheme $ELTA2E$ is a two-party lossy threshold encryption scheme secure against erasure-free one-sided active adaptive adversaries.*

Proof. By Lemma 2, $ELTA2E$ satisfies the lossy encryption properties. By Lemma 3, $ELTA2E$ satisfies the threshold semantic security requirement given in Definition 2. Then, $ELTA2E$ is a two-party lossy threshold encryption scheme secure against erasure-free one-sided active adaptive adversaries.

7 Oblivious Transfer Against One-Sided Active Adaptive Adversaries

In this section, a new protocol Π_{OTAA} for bit OT is presented. Let $\mathcal{F}_{OT}$ denote the ideal functionality for OT. Let $\mathcal{F}_{zk}$ denote the ideal functionality for adaptive zero-knowledge argument. The protocol Π_{OTAA} is presented below.

Protocol Π_{OTAA}.

CRS: $\mu \xleftarrow{\$} \mathbb{Z}_p$.
Input of S : $(x_0, x_1) \in \{0,1\}^2$.
Input of R : $\sigma \in \{0,1\}$.
Auxiliary Input: (n, p, q, g) where n is the security parameter, and (p, q, g) is the representation of a group $\mathbb{G}$ for the encryption scheme $ELTA2E = (K, KG, E, \Pi_{DEC})$.

1. S and R jointly generate an injective key for $ELTA2E$, by executing $\mathcal{F}_{KG}$ with input $(1^n, 1)$. Here, S and R acts as P_1 and P_2, respectively. Both parties get the public key $pk = (q, g, j, h, \ell)$, and the verification keys (vk, vk_1, vk_2). S gets its secret key share sk_1 and R gets its secret key share sk_2.
2. R selects $s_0, t_0, s_1, t_1 \xleftarrow{\$} \mathbb{Z}_q$. R computes $c_0 = E_{pk}\left(1 - \sigma, (s_0, t_0)\right)$, and $c_1 = E_{pk}\left(\sigma, (s_1, t_1)\right)$. R sends (c_0, c_1).
3. R proves that one of (c_0, c_1) is an encryption of zero, without disclosing which one. If R fails, then S aborts.
4. For each $i \in \{0,1\}$, S and R perform the following steps.
   (a) S computes $d_i = x_i \times_h c_i$. S computes $v_i = Blind(pk, d_i)$. S sends v_i.
   (b) S proves correctness of homomorphic multiplication by constant. If S fails, then R aborts.
5. For each $i \in \{0,1\}$, S and R jointly perform private decryption of v_i to R, as follows.
   (a) Let $v_i = (y_i, z_i)$. S sends $ds_{1,i} = (y_i)^{(sk_1)}$.
   (b) S proves that $\log_{(y_i)}(ds_{1,i}) = \log_{(vk)}(vk_1)$. If S fails, then R aborts.
   (c) R performs the following steps.

i. R computes $ds_{2,i} = (y_i)^{(sk_2)}$.
ii. R computes $\theta_i = \frac{z_i}{ds_{1,i} \cdot ds_{2,i}}$.
iii. From θ_i, R computes w_i where $w_i \in \{0,1\}$ and $g^{w_i} = \theta_i$ in $\mathbb{G}$.

6. R outputs w_σ.

Π_{OTAA} works in the CRS model in the $(\mathcal{F}_{zk}, \mathcal{F}_{KG})$-hybrid world. One possibility to generate the auxiliary inputs p, q, g is as follows. S generates the description (p, q, g) of the group $\mathbb{G}$ for $ELTA2E$, using Bach's algorithm [2]. S sends (p, q, g) to R. R checks its validity. If the description is invalid, then S and R repeat the same process. The proofs in steps 3, 4(b) and 5(b) are performed by adaptive zero-knowledge arguments. The CRS μ acts as the CRS for functionality $\mathcal{F}_{KG}$ and all the zero-knowledge arguments. In step 3, R proves that one of c_0 and c_1 encrypts zero, without disclosing which one. If R could set both ciphertexts c_0 and c_1 to encryptions of one, then R could learn both x_0 and x_1 at step 5. This proof is incorporated to prevent this type of cheating by R. In step 4, S computes $d_i = x_i \times_h c_i, v_i = Blind(d_i)$. S sends v_i. R knows the ciphertext c_i. The *Blind* function is included so that new randomness is added to the result d_i. Then, R cannot learn the constant x_i after seeing v_i.

Correctness of Protocol Π_{OTAA}. If S and R both follow the protocol, then the following events occur. S and R generate an injective key for $ELTA2E$. R honestly computes c_0 and c_1. c_σ encrypts one, and $c_{1-\sigma}$ encrypts zero. R passes the proof in step 3. S honestly performs step 4, and passes the proofs. v_σ encrypts x_σ and $v_{1-\sigma}$ encrypts zero. In step 5, S and R honestly perform two private decryption processes. By the "correctness on injective keys" property of $ELTA2E$, $w_\sigma = x_\sigma$ and $w_{1-\sigma} = 0$. Therefore, R learns x_σ.

Extension to String OT. In a string OT, S has a pair of bit strings of length q as input: $(\overline{x_0}, \overline{x_1}) = \left(\left\{x_0^1, x_0^2, \ldots, x_0^q\right\}, \left\{x_1^1, x_1^2, \ldots, x_1^q\right\}\right)$. Here q is a polynomial of n. R has input $\sigma \in \{0,1\}$. The output of R is $\overline{x_\sigma} = \left\{x_\sigma^1, x_\sigma^2, \ldots, x_\sigma^q\right\}$ and S does not get any output. The bit OT protocol Π_{OTAA} is extended to a string OT protocol as follows. In step 4, for each $i \in \{0,1\}, j \in \{1,2,\ldots,q\}, S$ computes $v_i^j = x_i^j \times_h c_i$. In step 5, for each $i \in \{0,1\}, j \in \{1,2,\ldots,q\}$, S and R jointly perform private decryption of v_i^j to R, so R obtains result w_i^j. R outputs $\left\{w_\sigma^1, w_\sigma^2, \ldots, w_\sigma^q\right\}$.

8 Security of Protocol Π_{OTAA}

The following theorem describes the security of protocol Π_{OTAA}.

Theorem 3. *Assume that the DDH assumption holds and there exists a trapdoor commitment scheme and adaptive zero-knowledge arguments. Assume that there exists a two-party lossy threshold public key cryptosystem which is secure against erasure-free one-sided active adaptive adversaries, is additive homomorphic, randomizable, and has an efficient (polynomial-time) Opener algorithm. Then, protocol Π_{OTAA} is a protocol for oblivious transfer secure under sequential composition, in the presence of erasure-free one-sided active adaptive adversaries.*

Proof. Let $\mathcal{A}$ be an erasure-free one-sided active adaptive adversary and $\mathcal{Z}$ be the environment. Let $\mathcal{S}_{OT}$ be the simulator for protocol Π_{OTAA} for adversary $\mathcal{A}$ and environment $\mathcal{Z}$. The security is proved using the SIP technique. The full security proof is available in the full version [16]. The main intuition behind the security is described for two cases below. In both cases, at start, $\mathcal{S}_{OT}$ selects $\delta \xleftarrow{\$} \mathbb{Z}_q$, and sets the CRS to $\mu = g^{\delta}$. $\mathcal{S}_{OT}$ stores δ as the trapdoor of the commitment key μ.

Case 1: Security for the case where S is the SIP
In step 1, $\mathcal{S}_{OT}$ generates a lossy key pair of $ELTA2E$ as follows. $\mathcal{S}_{OT}$ selects $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_q$. $\mathcal{S}_{OT}$ sets $\alpha = (\alpha_1 + \alpha_2) \bmod q, h = g^{\alpha}$. $\mathcal{S}_{OT}$ selects $\gamma \xleftarrow{\$} \mathbb{Z}_q$. $\mathcal{S}_{OT}$ sets $j = g^{\gamma}$. $\mathcal{S}_{OT}$ selects $\rho \xleftarrow{\$} \mathbb{Z}_q \setminus \{\alpha\}$, and sets $\ell = g^{\gamma\rho}$. $\mathcal{S}_{OT}$ sets $pk = (q, g, j, h, \ell)$. $\mathcal{S}_{OT}$ stores the corresponding secret key $sk = (\alpha, \gamma, \rho)$. $\mathcal{S}_{OT}$ generates the lossy key pair in a similar way to the way the key generation algorithm KG of $ELTA2E$ generates a lossy key pair. That means, the distribution of the key pair (pk, sk) is identically distributed to a lossy key pair generated by algorithm KG. The reason for generating the components of the keys, without using algorithm KG is as follows. When $\mathcal{S}_{OT}$ generates the values, it can obtain the values of α, γ and ρ. The secret key $sk = (\alpha, \gamma, \rho)$ is necessary to use the efficient *Opener* algorithm of $ELTA2E$. If protocol Π_{DKG} is used to implement step 1, then $\mathcal{S}_{OT}$ uses the simulator $\mathcal{S}_{DKG}$ of protocol Π_{DKG} on input $(pk, 0, P_1)$. That means $\mathcal{S}_{OT}$ invokes simulator $\mathcal{S}_{DKG}$ on input public key pk, mode parameter set to zero to denote lossy mode, and the identity I of the SIP set to P_1. By Theorem 1, the message that $\mathcal{S}_{DKG}$ generates in the hybrid world is computationally indistinguishable from the message that $\mathcal{A}$ views during the execution of Π_{DKG} in the real world. In the real world, an injective key pair is used. Since $\mathcal{A}$ corrupts at most one party, $\mathcal{A}$ cannot learn the secret key shares of both parties. So, $\mathcal{A}$ cannot learn the secret key. By the "indistinguishability of keys" property of $ELTA2E$, the public key in the hybrid world is computationally indistinguishable from the public key in the real world. If R is honest, then $\mathcal{S}_{OT}$ computes c_0, c_1 based on $\sigma = 0$ in step 2. By threshold semantic security of $ELTA2E$, the distribution of c_0, c_1 in two worlds are computationally indistinguishable. If $\mathcal{A}$ corrupts R after step 2, then $\mathcal{A}$ cannot replace the input σ as the value of σ is already fixed by the message supplied up to step 2. $\mathcal{S}_{OT}$ corrupts R in the hybrid world and receives its input σ from $\mathcal{Z}$. $\mathcal{S}_{OT}$ sends σ to the trusted party of $\mathcal{F}_{OT}$, and receives back its output x_{σ}. If R is corrupted, then $\mathcal{S}_{OT}$ acts as an honest verifier in step 3. If R fails, then $\mathcal{S}_{OT}$ sends $abort_R$ to the trusted party and halts. The trusted party sends $abort_R$ to S and S halts. In this case, honest S aborts in the real world. If R passes, then $\mathcal{S}_{OT}$ extracts the plaintexts of c_0 and c_1 by using the knowledge extractor of the zero-knowledge arguments. From these plaintexts, $\mathcal{S}_{OT}$ learns the possibly modified input σ_1 of corrupted R. $\mathcal{S}_{OT}$ sends σ_1 to the trusted party of $\mathcal{F}_{OT}$, and receives back its output x_{σ_1}. $\mathcal{S}_{OT}$ sets $\sigma = \sigma_1$ and the output of R to x_{σ_1}. In the real world, the generated key pair is injective, so $\mathcal{A}$ cannot open a ciphertext encrypting one to be a ciphertext encrypting zero. In the hybrid world, $\mathcal{S}_{OT}$ generates a lossy key pair. Since $\mathcal{A}$ corrupts at most one party, $\mathcal{A}$ cannot learn

the secret key. Without the knowledge of the secret key, $\mathcal{A}$ cannot use the efficient *Opener* algorithm as the efficient *Opener* algorithm requires the secret key as one of its inputs. That means in the hybrid world, $\mathcal{A}$ cannot open a ciphertext encrypting one to be a ciphertext encrypting zero in polynomial time. That means the result of the zero-knowledge argument will be identical in both worlds. If $\sigma = 0$, then $\mathcal{S}_{OT}$ performs no additional updates in step 3, since $\mathcal{S}_{OT}$ calculated c_0, c_1 based on $\sigma = 0$. If $\sigma = 1$, then, in step 3, $\mathcal{S}_{OT}$ computes randomness $\hat{s}_0, \hat{t}_0, \hat{s}_1, \hat{t}_1$ using the efficient *Opener* algorithm, such that $c_0 = E_{pk}(0, (\hat{s}_0, \hat{t}_0))$ and $c_1 = E_{pk}(1, (\hat{s}_1, \hat{t}_1))$. $\mathcal{S}_{OT}$ supplies $\hat{s_0}, \hat{t_0}, \hat{s_1}, \hat{t_1}$ as the randomness for step 2. Since $\mathcal{S}_{OT}$ knows the secret key of the lossy key pair, algorithm *Opener* produces output in polynomial time. By the "openability" property of $ELTA2E$, the generated randomness is consistent. In step 4(a), $\mathcal{S}_{OT}$ selects $\hat{x}_i \xleftarrow{\$} \{0,1\}$, computes $d_i = \hat{x}_i \times_h c_i, \hat{v}_i = Blind(pk, d_i)$. By threshold semantic security of $ELTA2E$, the distribution of v_i in two worlds are computationally indistinguishable. Correctness of decryption does not hold for a lossy key for $ELTA2E$. So, $\mathcal{S}_{OT}$ sets $w_\sigma = x_\sigma, w_{1-\sigma} = 0$. $\mathcal{S}_{OT}$ computes $\theta_i = g^{w_i}, ds_{2,i} = (vy_i)^{sk_2}, \widehat{ds}_{1,i} = \frac{vz_i}{\theta_i \cdot ds_{1,i}}$. In the real world, $\mathcal{A}$ receives $ds_{1,i} = (y_i)^{sk_1}$. Since S is honest, so $\mathcal{A}$ does not know sk_1. By the DDH assumption, the distribution of $ds_{1,i}$ in two worlds are computationally indistinguishable. The proofs of step 3 and step 4(b) do not work for a lossy key for $ELTA2E$. If R is honest, then $\mathcal{S}_{OT}$ generates a proof transcript for steps 3, 4(b), and 5(b) using the trapdoor δ. By definition of zero-knowledge argument, the proof transcripts in two worlds are computationally indistinguishable. If R is honest, then $\mathcal{S}_{OT}$ honestly performs step 5(c). If R is corrupted, then, in the hybrid world, $\mathcal{A}$ obtains w_i. In the real world, $\mathcal{A}$ obtains w_i due to the "correctness on injective keys" property of $ELTA2E$. If R is corrupted, then $\mathcal{A}$ will obtain the same value x_σ in step 6 in the hybrid world that it obtains in the real world. In an OT protocol, S has no output. So trivially, the output of the honest party S is identical (an empty string) in both worlds. If $\mathcal{A}$ corrupts R after any substep of step 4 or 5, then $\mathcal{S}_{OT}$ performs the same steps if $\mathcal{A}$ corrupts R after step 3.

Case 2: Security for the case where R is the SIP
In step 1, $\mathcal{S}_{OT}$ performs similar to step 1 in case 1. If Π_{DKG} is used to generate the key, then, $\mathcal{S}_{OT}$ uses the simulator $\mathcal{S}_{DKG}$ of protocol Π_{DKG} on input $(pk, 0, P_2)$. $\mathcal{S}_{OT}$ computes c_0, c_1 based on $\sigma = 0$ in step 2. Proof argument is similar to step 2 of case 1. In step 4, if S is honest, then $\mathcal{S}_{OT}$ selects $\hat{x}_i \xleftarrow{\$} \{0,1\}$, computes $\hat{d}_i = \hat{x}_i \times_h c_i, \hat{v}_i = Blind(\hat{d}_i)$. Proof argument is similar to step 4 of case 1. If S is corrupted, then $\mathcal{S}_{OT}$ acts as an honest verifier in steps 4(b) and 5(b). If S fails in any proof, then $\mathcal{S}_{OT}$ sends $abort_S$ to the trusted party and halts. In this case, honest R aborts in the real world. If S passes, then $\mathcal{S}_{OT}$ extracts the possibly replaced input $\tilde{x}_i$ by using the knowledge extractor of the zero-knowledge argument. If S is honest, then $\mathcal{S}_{OT}$ generates proof transcripts for steps 4(b) and 5(b) using trapdoor δ. By definition of zero-knowledge argument, the proof transcript in two worlds are computationally indistinguishable. If $\mathcal{A}$ corrupts S after step 4, then $\mathcal{S}_{OT}$ corrupts S in the hybrid world, and receives its input (x_0, x_1) from $\mathcal{Z}$. In this case,

$\mathcal{A}$ cannot replace the input (x_0, x_1) as the value of (x_0, x_1) is already fixed by the message sent up to step 4. $\mathcal{S}_{OT}$ sets $(\tilde{x}_0, \tilde{x}_1)$ to (x_0, x_1). If $\hat{x_i} \neq \tilde{x}_i$, then $\mathcal{S}_{OT}$ computes randomness for the ciphertexts transmitted so far and the value of $\tilde{x}_i$, using the efficient *Opener* algorithm. By the "openability" property of $ELTA2E$, the randomness generated is consistent. Since $\mathcal{S}_{OT}$ knows the secret key, the *Opener* algorithm produces output in polynomial-time. $\mathcal{S}_{OT}$ sets $w_\sigma = \tilde{x}_\sigma, w_{1-\sigma} = 0$. If S is honest, then $\mathcal{S}_{OT}$ computes $\theta_i = g^{w_i}, ds_{2,i} = (y_i)^{sk_2}, \widehat{ds}_{1,i} = \frac{z_i}{\theta_i \cdot ds_{2,i}}$. In step 6, $\mathcal{S}_{OT}$ sends $(\tilde{x}_0, \tilde{x}_1)$ to the trusted party of $\mathcal{F}_{OT}$. Let σ be the input of honest R. Then the trusted party sends the output $\tilde{x}_\sigma$ to R. In the real world, honest R outputs the value $\tilde{x}_\sigma$, by the "correctness on injective keys" property of $ELTA2E$. Then, the output of the honest party R is identical in two worlds. If $\mathcal{A}$ corrupts R after any substep of step 3, 4 or 5, then $\mathcal{S}_{OT}$ performs the same steps if $\mathcal{A}$ corrupts R after step 3(a).

9 Efficiency and Comparison with Related Work

Efficiency. In $ELTA2E$, a ciphertext $c \in \mathbb{G} \times \mathbb{G}$. $\mathbb{G}$ is a subgroup of $\mathbb{Z}_p^*$ and p is an n-bit prime. The size of ciphertext is $2n$. The size of Pedersen commitment [18] is n. It is possible to use protocol Π_{DKG} for implementing step 1 of Π_{OTAA}. The communication complexity of Π_{DKG} is $50n$. The total communication complexity of Π_{OTAA} (including the communication complexity of Π_{DKG}) is $101n \in O(n)$. In step 2, R performs two encryption operations of $ELTA2E$. In step 5, S performs two homomorphic multiplication by constant and two *Blind* function evaluations. One homomorphic multiplication by constant and one *Blind* function together is similar in computational complexity of one encryption operation of $ELTA2E$. So, the total number of PKE operation of Π_{OTAA} is 4, in the worst case. For string OT of length n, the communication complexity is $(38n^2 + 98n)$, and the number of PKE operations is $(2n + 2)$, in the worst case.

Comparison with Related Work. Hazay and Patra [13] designed an OT protocol for erasure-free one-sided active adaptive adversaries. Their protocol for bit OT requires $(288n^2 + 100n + 16) \in O(n^2)$ communication complexity. Protocol Π_{OTAA} needs $101n \in O(n)$ communication complexity. The worst case number of PKE operations of the protocol of [13] for bit OT is $(16n+6) \in O(n)$. The worst case number of PKE operations of Π_{OTAA} is constant (only four).

For OT of strings of size n, the OT protocol of [13] requires $(288n^2+110n+16)$ communication complexity and $(16n+6)$ PKE operations in the worst case. For OT of strings of size n, protocol Π_{OTAA} requires $(38n^2 + 98n)$ communication complexity and $(2n + 2)$ PKE operations in the worst case. For string OT of size n, protocol Π_{OTAA} requires seven times less communication complexity and eight times less PKE operations than the OT protocol of [13].

10 Efficiency of the OT Protocol by Hazay and Patra [13]

In this section, the main factor of the complexity of the OT protocol by Hazay and Patra [13] is described. They have different efficiency for polynomial-size message space and exponential-size message space, with respect to the security parameter n. Here, the efficiency of bit OT, which falls in the category of polynomial-size message space, is described.

The OT protocol of [13] uses a non-committing encryption (NCE) scheme secure against one-sided active adaptive adversaries. They designed a protocol Π_{OSC} that for this purpose. Π_{OSC} uses the somewhat non-committing encryption (SNCE) of [12]. The SNCE protocol of [12] uses the non-committing encryption scheme (NCE) of [10]. There was another more recent NCE scheme [7] which is error-free but requires more communication complexity and computational complexity than the NCE of [10,21]. The NCE scheme of [10] uses a subroutine named *attempt*. In [Theorem 2 [10]], it is mentioned that the NCE scheme of [10] has to repeat $4n$ calls of *attempt* in order to ensure that the probability of failure of subroutine *attempt* remains negligible in n. That means, the worst case number of repeats of *attempt* is $4n$. Each call of *attempt* has communication cost $(12n + 1)$. The communication complexity of the NCE scheme of [10] is $O(n^2)$ for message size of one bit. Each call of *attempt* uses one encryption operation of a simulatable public key encryption scheme, so the number of PKE operations for *attempt* is 1. Then, the NCE scheme of [10] needs $4n$ PKE operations in the worst case. The communication complexity of the SNCE protocol of [12], with equivocality parameter $\ell = 2$, is $O(n^2)$. It uses the NCE protocol of [10] for sending an index $i \in \{1, \ldots, \ell\}$. As mentioned in [12], the expected number of PKE operations for this step is $O(\log \ell)$. In the worst case, this step requires $4n \in O(n)$ PKE operations. The OT protocol of [13] uses a zero-knowledge proof that uses a constant number of PKE operations. The communication complexity of the bit OT protocol of [13] is $O(n^2)$. The number of PKE operations of the bit OT of [13], in the worst case, is $O(n)$.

Hazay and Patra claims that their OT protocol needs a constant number of PKE operations [Theorem 2 [13]]. One possibility is that they were counting one encryption of the NCE scheme Π_{OSC} secure against one-sided adaptive adversaries (or one encryption of the dual-mode encryption scheme of [19]), each of them as a single PKE operation. But the encryption scheme Π_{OSC} uses other PKE schemes (the non-committing encryption scheme for the sender (NCES) of [3], the non-committing encryption scheme for the receiver (NCER) of [14] and the SNCE scheme of [12]) as its subroutines inside its implementation. The notion of *atomic* PKE scheme is necessary for the analysis of the number of PKE operations. An *atomic* PKE scheme denotes a PKE scheme that does not use any other PKE scheme as a subroutine in its implementation. To get the actual number of PKE scheme of a protocol, it should be counted that how many operations of atomic PKE scheme are invoked inside that protocol.

11 Adaptive Zero Knowledge Arguments

In this section, the adaptive zero-knowledge arguments used in the protocols are described. First, the non-erasure Σ-protocols for the corresponding relations are presented. Then, it is described how to convert them to adaptive zero-knowledge arguments.

Scnorr [20] suggested a non-erasure Σ-protocol for knowledge of discrete logarithm [8]. A non-erasure Σ-protocol for equality of discrete logarithm is given in [8]. Damgård [8] presented a non-erasure Σ-protocol for proving knowledge of committed secret for Pedersen commitment scheme.

If $mode = 0$, then, in step 10 of Π_{DKG}, P_1 has to prove that $\log_j(\ell_1) \neq \log_{vk}(vk_1)$. This can be called a proof for inequality of discrete logarithm. Let the common input be $(x_1, x_2, y_1, y_2) = ((y_1)^{w_1} \bmod p, (y_2)^{w_2} \bmod p, y_1, y_2)$. The prover P knows witness $w_1, w_2 \in \mathbb{Z}_q$ such that $w_1 \neq w_2$. A new non-erasure Σ-protocol for proving the inequality of discrete logarithm is designed. P chooses $r \xleftarrow{\$} \mathbb{Z}_q$. P computes $a_1 = (y_1)^r \bmod p, a_2 = (y_2)^r \bmod p$. P sends $a = (a_1, a_2)$. V chooses a challenge $e \xleftarrow{\$} \mathbb{Z}_q$ and sends it. P computes $z_1 = r + ew_1 \bmod q, z_2 = r + ew_2 \bmod q$. P sends (z_1, z_2). V accepts if and only if $(y_1)^{z_1} = a_1(x_1)^e \bmod p, (y_2)^{z_2} = a_1(x_2)^e \bmod p, (y_1)^{z_2} \neq a_1(x_1)^e \bmod p$, and $(y_2)^{z_1} \neq a_2(x_2)^e \bmod p$.

A new non-erasure Σ-protocol, for proving multiplication correct for $ELTA2E$, is designed. P computes $c_2 = m_2 \times_h c_1, c_3 = Blind(pk, c_2)$. Let (s_3, t_3) be the randomness used in the $Blind$ function. Let the common input be $x = (c_1, c_3) = ((b_1, d_1), (b_3, d_3))$. Then, $b_3 = (b_1)^{m_2} g^{s_3} j^{t_3}, d_3 = (d_1)^{m_2} h^{s_3} \ell^{t_3}$. P knows witness $(m_2, s_3, t_3) \in \mathbb{G} \times \mathbb{G} \times \mathbb{G}$. The Σ-protocol is as follows. P chooses $r_1, r_2, r_3 \xleftarrow{\$} \mathbb{Z}_q$, sets $a_1 = (b_1)^{r_1} g^{r_2} j^{r_3} \bmod p, a_2 = (d_1)^{r_1} h^{r_2} \ell^{r_3} \bmod p$. P sends $a = (a_1, a_2)$. V chooses a challenge $e \xleftarrow{\$} \mathbb{Z}_q$ and sends it. P sets $z_1 = r + em_2 \bmod q, z_2 = r_2 + es_3 \bmod q, z_3 = r_3 + et_3 \bmod q$. P sends $z = (z_1, z_2, z_3)$. V accepts if and only if $(b_1)^{z_1} g^{z_1} j^{z_3} = a_1(b_3)^e \bmod p$, and $(d_1)^{z_1} h^{z_1} \ell^{z_3} = a_2(d_3)^e \bmod p$.

Proving that a given ciphertext $c = (x, y)$ is an encryption of zero is equivalent to prove that $\log_g(x) = \log_h(y)$. For proving that one of two given ciphertexts encrypts zero without disclosing which one, the OR-construction of Σ-protocols [8] is performed.

Converting Σ-Protocol to Adaptive Zero-Knowledge Argument. Damgård [9] described how to convert a Σ-protocol $\Pi_{\Sigma R}$ for a given relation R to a zero-knowledge proof Π^R_{AdZKA} for the same relation. This conversion works in the CRS model and needs a trapdoor commitment scheme. The CRS μ is used as the commitment key. P computes its first message a of $\Pi_{\Sigma R}$, selects $r_a \xleftarrow{\$} \mathbb{Z}_p$, computes $c = Com_\mu(a, r_a)$, and sends c. V selects $e \xleftarrow{\$} \{0,1\}^t$, and sends e. P computes its second message z of $\Pi_{\Sigma R}$, and sends (a, z, r_a) to V. V checks that (a, e, z) is an accepting conversation for $\Pi_{\Sigma R}$, and also that $Com_\mu(a, r_a) = c$. The security proof of this type of zero-knowledge proof against adaptive adversaries is given in [[17] Chap. 5]. When a trapdoor commitment scheme is used in

a zero-knowledge proof, it only achieves computational soundness. By definition, the resulting system is a zero-knowledge argument.

12 Future Work

One future research work is to design an efficient two-party computation protocol for one-sided active adaptive adversary model, using the new efficient oblivious transfer protocol. Another research direction is to design efficient oblivious transfer protocol for the fully adaptive adversary model, that is, when the adversary may corrupt both parties at some point.

References

1. Aumann, Y., Lindell, Y.: Security against covert adversaries: efficient protocols for realistic adversaries. J. Cryptol. **23**(2), 281–343 (2010)
2. Bach, E.: Analytic Methods in the Analysis and Design of Number-Theoretic Algorithms. Massachusetts Institute of Technology, Cambridge (1985)
3. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and impossibility results for encryption and commitment secure under selective opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (2009). doi:10.1007/978-3-642-01001-9_1
4. Bellare, M., Yilek, S.: Encryption schemes secure under selective opening attack. Cryptology ePrint Archive, Report 2009/101 (2009). http://eprint.iacr.org/
5. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1), 143–202 (2000)
6. Canetti, R., Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Adaptive security for threshold cryptosystems. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 98–116. Springer, Heidelberg (1999). doi:10.1007/3-540-48405-1_7
7. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved non-committing encryption with applications to adaptively secure protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 287–302. Springer, Heidelberg (2009). doi:10.1007/978-3-642-10366-7_17
8. Damgård, I.: On Σ-protocols. www.cs.au.dk/~ivan/Sigma.pdf
9. Damgård, I.: Efficient concurrent zero-knowledge in the auxiliary string model. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 418–430. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_30
10. Damgård, I., Nielsen, J.B.: Improved non-committing encryption schemes based on a general complexity assumption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 432–450. Springer, Heidelberg (2000). doi:10.1007/3-540-44598-6_27
11. Fouque, P.-A., Poupard, G., Stern, J.: Sharing decryption in the context of voting or lotteries. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 90–104. Springer, Heidelberg (2001). doi:10.1007/3-540-45472-1_7
12. Garay, J.A., Wichs, D., Zhou, H.-S.: Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 505–523. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03356-8_30

13. Hazay, C., Patra, A.: One-sided adaptively secure two-party computation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 368–393. Springer, Heidelberg (2014). doi:10.1007/978-3-642-54242-8_16
14. Jarecki, S., Lysyanskaya, A.: Adaptively secure threshold cryptography: introducing concurrency, removing erasures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 221–242. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_16
15. Naor, M., Ostrovsky, R., Venkatesan, R., Yung, M.: Perfect zero-knowledge arguments for NP using any one-way permutation. J. Cryptol. **11**, 87–108 (1998)
16. Nargis, I.: Efficient oblivious transfer from lossy threshold homomorphic encryption. Cryptology ePrint Archive, Report 2017/235 (2017). http://eprint.iacr.org/2017/235
17. Nielsen, J.B.: On protocol security in the cryptographic model. Ph.D. thesis, University of Aarhus (2004)
18. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992). doi:10.1007/3-540-46766-1_9
19. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85174-5_31
20. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991)
21. Zhu, H., Araragi, T., Nishide, T., Sakurai, K.: Adaptive and composable non-committing encryptions. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 135–144. Springer, Heidelberg (2010). doi:10.1007/978-3-642-14081-5_9