Efficiency-Improved Fully Simulatable Adaptive OT under the DDH Assumption

Kaoru Kurosawa[1], Ryo Nojima[2], and Le Trieu Phong[2]

[1] Ibaraki University, Japan, kurosawa@mx.ibaraki.ac.jp
[2] NICT, Japan, {ryo-no, phong}@nict.go.jp

Abstract. At Asiacrypt 2009, Kurosawa and Nojima showed a fully simulatable adaptive oblivious transfer (OT) protocol under the DDH assumption in the standard model. However, Green and Hohenberger pointed out that the communication cost of each transfer phase is $O(n)$, where n is the number of the sender's messages. In this paper, we show that the cost can be reduced to $O(1)$ by utilizing a verifiable shuffle protocol.

Keywords: Adaptive OT, verifiable shuffles, DDH, standard model.

1 Introduction

1.1 Background

Adaptive oblivious transfer is a notion introduced by Naor and Pinkas in [12]. In the scheme, denoted by $\mathrm{OT}^n_{k\times 1}$, a receiver can obtain k messages, *one after the other*, from a sender who has n messages in such a way that: (1) the sender learns nothing on the receiver's selection, and (2) the receiver only learns about the k messages. The key applications of this type of OT are in patent searches, oblivious search, medical databases etc.

The formal security definition for OT schemes capturing the above intuitions gets evolved in the literature. Historically, in half simulation security [14], only the sender security is defined via the real world/ideal world paradigm, while the receiver security is formalized by a *weaker* notion. Many OT schemes in the literature satisfy half simulation security, among which are [3, 9, 11, 13, 18]. However, there is a practical attack against schemes with half simulation security, as realized in [11] and formally emphasized in [1].

To overcome the threat, in 2007, Camenisch, Neven, and shelat introduced a stronger notion called "full simulation security" [1], in which *both* sender and receiver security are defined via the real world/ideal world paradigm. They then constructed a fully simulatable adaptive $\mathrm{OT}^n_{k\times 1}$ in the standard model, relying on the q-strong Diffie-Hellman (q-sDH) and q-power decisional Diffie-Hellman (q-PDDH) assumptions in bilinear groups. Camenisch, Neven, and shelat used signatures as a key ingredient in their approach, which was originally taken in [18] by Ogata and Kurosawa in the random oracle model.

Subsequently, in 2008, Green and Hohenberger, again using signatures, showed a universally composable scheme (and hence fully simulatable), relying on the

Table 1. Fully simulatable adaptive OT without random oracles

Scheme	Assumption	Comm. Cost (each transfer)
Camenisch et al [1]	q-strong DH and q-PDDH	$O(1)$
Green-Hohenberger [6]	q-hidden LRSW (UC secure)	$O(1)$
Jarecki-Liu [8]	q-DHI (RSA group)	$O(1)$
Kurosawa-Nojima [10]	DDH	$O(n)$
Green-Hohenberger [7]	decision 3-party DH (3DDH)	O(1)
This work	DDH	$O(1)$

q-hidden LRSW assumption. In 2009, Jarecki and Liu [8], using pseudorandom function as a component, presented a fully simulatable adaptive OT under the decisional q-Diffie-Hellman inversion (q-DHI) assumption.

We stress that all the above schemes rely on dynamic assumptions (namely, the q-based assumptions in Table 1 where q may depend on n, the number of messages in OT). In 2009, Kurosawa and Nojima [10] built a simple fully simulatable adaptive OT under the DDH assumption. However, Green and Hohenberger [7] pointed out that it has $O(n)$ communication cost in each transfer phase which is much larger than the other schemes. Green and Hohenberger [7] also also proposed a fully simulatable adaptive OT under the decision 3-party Diffie-Hellman (3DDH) assumption, with $O(1)$ communication cost in each transfer phase.

1.2 Our contribution

In this paper, we show a fully simulatable adaptive OT under the DDH assumption such that each transfer requires only $O(1)$ communication cost in the standard model. (The initialization phase requires $O(n)$ communication cost, which is asymptotically minimal.) Note that the DDH assumption is a more standard assumption than the 3DDH assumption on which the scheme of Green and Hohenberger [7] relies. Furthermore our scheme does not use pairing, while the scheme of Green and Hohenberger [7] does.

Our scheme is obtained by improving the scheme of Kurosawa and Nojima [10] by using a verifiable shuffle protocol. To our knowledge, this is the first time that shuffles are used in building OT protocols. In particular, we employ the shuffle protocol of Neff [16, 17] in this paper. The technique helps greatly reducing the communication cost of each transfer from $O(n)$ in the Kurosawa-Nojima scheme [10] to $O(1)$ as in our proposal.

A comparison between schemes is given in Table 1, and a motivation behind the usage of shuffles is postponed later in Sec.4.

Organization. We begin with some preliminaries in Sec.2, then introduce a verifiable shuffle protocol for our OT construction in Sec.3. We describe our proposal and prove its security in Sec.4.

2 Preliminaries

We will work on a cyclic group G of prime order q, generated by an element g. The symbol " $\xleftarrow{\$}$ " indicates a randomized process.

2.1 Assumption

The DDH assumption claims that for all PPT adversary $\mathcal{A}$, the value

$$\mathbf{Adv}_G^{ddh}(\mathcal{A}) = \left| \Pr \left[b' = b : \begin{array}{c} x, r \xleftarrow{\$} Z_q; b \xleftarrow{\$} \{0,1\}; \\ T_0 \leftarrow g^{xr}; T_1 \xleftarrow{\$} G; \\ b' \xleftarrow{\$} \mathcal{A}(g, g^x, g^r, T_b) \end{array} \right] - \frac{1}{2} \right|$$

is negligible. The well-known ElGamal encryption, which has semantic security under the DDH assumption, produces a ciphertext of a message $M \in G$ as $(g^r, M \cdot (g^x)^r)$ for public key g^x.

2.2 Zero-Knowledge Proof Systems

There exists an efficient 4-round zero-knowledge proof system for knowledge (ZK-PoK) on the discrete log problem. It is obtained by applying the technique of [4] to Schnorr's identification scheme [19].

There also exists an efficient 4-round zero-knowledge proof system for membership (ZK-PoM) on DDH tuples (i.e., $(g, g^x, u, u^x) \in G^4$). It comes from the confirmation protocol of Chaum's undeniable signature scheme [2].

2.3 Security of Adaptive k-out-of-n Oblivious Transfer

We use almost the same presentation as [10], and consider a weak model of universally composable (UC) framework as follows.

- At the beginning of the game, an adversary $\mathcal{A}$ can corrupt either a sender S or a receiver R, but not both of them.
- $\mathcal{A}$ can send a message, denoted by $\mathcal{A}_{\text{out}}$, to an environment $\mathcal{Z}$ after the end of the protocol. However, $\mathcal{A}$ cannot communicate with $\mathcal{Z}$ during the protocol execution. (This property makes the definitions weaker than standard UC security.)

The ideal functionality of $\mathrm{OT}_{k\times 1}^n$ will be shown below. For a protocol $\Pi = (\mathsf{S}, \mathsf{R})$, define the advantage of $\mathcal{Z}$ as

$$\mathbf{Adv}(\mathcal{Z}) \stackrel{\text{def}}{=} \Big| \Pr(\mathcal{Z} = 1 \text{ in the real world}) - \Pr(\mathcal{Z} = 1 \text{ in the ideal world}) \Big|$$

where the real and ideal worlds are defined below.

In the ideal world of $\mathrm{OT}_{k\times 1}^n$, there are a few parties: the ideal functionality $\mathcal{F}_{\text{adapt}}$, an ideal world adversary $\mathcal{A}'$, and the environment $\mathcal{Z}$. Also we have dummy sender S' and receiver R'. The parties behave as follows.

Initialization phase

1. The environment $\mathcal{Z}$ sends $(M_1, \ldots, M_n)$ to the dummy sender S'.
2. S' sends $(M_1^*, \ldots, M_n^*)$ to $\mathcal{F}_{\text{adapt}}$, where $(M_1^*, \ldots, M_n^*) = (M_1, \ldots, M_n)$ if S' is not corrupted.

Transfer phase $i = 1, \ldots, k$

1. $\mathcal{Z}$ sends σ_i to the dummy receiver R', where $1 \leq \sigma_i \leq n$.
2. R' sends σ_i^* to $\mathcal{F}_{\text{adapt}}$, where $\sigma_i^* = \sigma_i$ if R' is not corrupted.
3. $\mathcal{F}_{\text{adapt}}$ sends `received` to $\mathcal{A}'$.
4. $\mathcal{A}'$ sends $b = 1$ or 0 to $\mathcal{F}_{\text{adapt}}$, where $b = 1$ if S' is not corrupted.
5. $\mathcal{F}_{\text{adapt}}$ sends E_i to R', where

$$E_i = \begin{cases} M_{\sigma_i^*}^* & \text{if } b = 1 \\ \perp & \text{if } b = 0 \end{cases}$$

6. R' sends E_i to $\mathcal{Z}$.

After the end of the protocol, $\mathcal{A}'$ sends a message $\mathcal{A}'_{\text{out}}$ to $\mathcal{Z}$. Finally $\mathcal{Z}$ outputs 1 or 0.

On the other hand, in the real world, the protocol $\Pi = (\mathsf{S}, \mathsf{R})$ is executed as specified by its construction (thus without $\mathcal{F}_{\text{adapt}}$). The environment $\mathcal{Z}$ and the real world adversary $\mathcal{A}$ behave in the same way as above.

Definition 1. *Protocol $\Pi = (\mathsf{S}, \mathsf{R})$ is secure against the sender (resp, receiver) corruption if for any real world adversary $\mathcal{A}$ who corrupts the sender S (resp, receiver R), there exists an ideal world adversary $\mathcal{A}'$ who corrupts the dummy sender S' (resp, dummy receiver R') such that for any poly-time environment $\mathcal{Z}$, the advantage $\mathbf{Adv}(\mathcal{Z})$ is negligible.*

Definition 2. *Protocol $\Pi = (\mathsf{S}, \mathsf{R})$ is a fully simulatable $OT_{k\times 1}^n$ if it is secure against the sender corruption and the receiver corruption.*

3 Shuffle Protocol

3.1 Honest-Verifier ZK-PoM

Neff [16, Sec.5] showed a seven-round ZK-PoM on L where

$$L = \{(g, g^c, X_1, \ldots, X_n, X_{\pi(1)}^c, \ldots, X_{\pi(n)}^c \mid c \in Z_q,\ \pi \text{ is a permutation on } \{0,1\}^n\}$$

Note that we can extract π if we know c.

It is easy to see that $(g, g^c, X_1, \ldots, X_n, X_1^c, \ldots, X_n^c)$ is indistinguishable from $(g, g^c, X_1, \ldots, X_n, R_1, \ldots, R_n)$ under the DDH assumption, where $R_1, \ldots, R_n$ are random elements of G. This implies that $(g,\ g^c,\ X_1,\ \ldots,\ X_n,\ X_{\pi(1)}^c,\ \ldots,\ X_{\pi(n)}^c)$ leaks no information on π computationally. Formalizing the intuition, Neff proved that his proof system is honest-verifier computational zero-knowledge under the DDH assumption. The communication cost for the proof system is $O(n)$.

3.2 Any Verifier ZK-PoM

The above protocol (P, V) of Neff is public coin. That is, V sends random elements of Z_q to P. We can transform it into an any verifier ZK-PoM by having V commit the random elements at the beginning of the protocol. (By using the same technique, Goldreich and Kahan [5] showed a constant round ZK-PoM for any NP language under the discrete log assumption. However, as a trade-off against the generality, their protocol is very inefficient.)

For example, suppose that V sends a random $t \in Z_q$ to P in the first round of (P, V). Then we transform it as follows.

1. P sends a random $h \in G$ to V.
2. V chooses random $t_0, r \in Z_q$, and computes

$$\texttt{commit}(t_0, r) = g^{t_0} h^r. \tag{1}$$

   He then send it to P.
3. P sends a random $t_1 \in Z_q$ to V.
4. V reveals t_0 and r.
5. If eq.(1) is not satisfied, then P aborts. Otherwise P and V computes

$$t = t_0 + t_1 \bmod q$$

   locally.

As a result, we obtain a constant round ZK-PoM on L with respect to any verifier. It is computational zero-knowledge under the DDH assumption. The communication cost is still $O(n)$.

3.3 An Alternative Shuffle Protocol

The verifiable shuffle protocol described in Sec.3.1 is for honest verifier, and as mentioned above, needs a conversion to the case of any verifier. We provide in this section an alternative shuffle protocol which is zero-knowledge, under the DDH assumption, with respect to any verifier for the language L without the above conversion. As a consequence, we obtain a 7-round zero knowledge shuffle protocol with respect to *any* verifier.

1. For $i = 1, \ldots, k$, P chooses $a_i \in Z_q$ randomly and sends $A_i = g^{a_i}$ to V.
2. For $i = 1, \ldots, k$, V sends a random $b_i \in Z_q$ to P.
   P and V compute $B_i = A_i g^{b_i}$ locally.
3. For $i = 1, \ldots, k$, P computes

$$\begin{aligned} C_i &= B_{\pi(i)}^{c} \\ \bar{X}_i &= X_i^{a_i + b_i} \\ \bar{Y}_i &= Y_i^{c(a_{\pi(i)} + b_{\pi(i)})}, \end{aligned}$$

where $Y_i = X_{\pi(i)}$. Also P computes

$$U = \left(\prod_{i=1}^{k} \bar{X}_i \right)^c$$

and sends

$$U, (C_1, \ldots, C_k), (\bar{X}_1, \ldots, \bar{X}_k), (\bar{Y}_1, \ldots, \bar{Y}_k)$$

to V.
P and V compute

$$S = \prod_{i=1}^{k} \bar{X}_i \text{ and } T = \prod_{i=1}^{k} \bar{Y}_i$$

locally.

4. P and V run the simple k-shuffle protocol [16, Sec.4] for

$$(B_1, \ldots, B_k),\ (C_1, \ldots, C_k),$$

in which P is required to know $\log_g(B_i)$ and $\log_g(C_i)$ for all i, a condition which is obviously fulfilled.
For $i = 1, \ldots k$, P proves that $(g, X_i, B_i, \bar{X}_i)$ and $(g, Y_i, C_i, \bar{Y}_i)$ are DDH tuples [2].
P also proves that (g, g^c, S, U) and (g, g^c, U, T) are DDH tuples.

4 Proposed Adaptive OT under DDH Assumption

In this section, we show an efficient fully simulatable adaptive $\mathrm{OT}^n_{k\times 1}$ under the DDH assumption. Each transfer phase needs only $O(1)$ communication cost, and the initialization phase requires $O(n)$ communication cost.

The novelty of our protocol is that we use a shuffle protocol in the initialization phase. Namely we use the ZK-PoM shown in Sec.3.2. A problem is that since it is not a ZK-PoK, we cannot extract π from the prover. This problem is solved by having the prover run the ZK-PoK in which P proves that she knows c of g^c. Then π can be extracted from c and $(X_1, \ldots, X_n, X^c_{\pi(1)}, \ldots, X^c_{\pi(n)})$.

4.1 Protocol

As a convention, if proofs or checks are not fulfilled, it is implicitly understood that the protocol immediately stops.

Initialization Phase

1. The sender chooses $(r_1, \ldots, r_n, x) \in Z_q^{n+1}$ randomly, and computes $h = g^x$.
2. For $i = 1, \ldots, n$, the sender computes

$$C_i = (A_i, B_i) = (g^{r_i}, M_i \cdot h^{r_i}),$$

where $M_1, \ldots, M_n \in G$.

3. The sender sends $(h, C_1, \ldots, C_n)$.
4. The sender proves by ZK-PoK that he knows the secret key x.
5. The receiver chooses $c \in Z_q$ and sends $C = g^c$. Then he proves in ZK-PoK that he knows c.
6. The receiver chooses $s_i \in Z_q$ randomly and computes $X_i = g^{s_i} A_i$ for every $1 \leq i \leq n$. He sends all X_i and then proves in ZK-PoK that he knows s_i for every i.
7. (Shuffling) The receiver chooses a random permutation π on $\{1, \ldots, n\}$. Then he sends
$$(Y_1, \ldots, Y_n) \stackrel{\text{def}}{=} (X^c_{\pi(1)}, \ldots, X^c_{\pi(n)}).$$
He proves that there exist such π and c by using the ZK-PoM of Sec.3.2. The communication cost is $O(n)$.

The j-th Transfer Phase

1. The receiver obtains an index $1 \leq \sigma \leq n$.
2. The receiver sends $U = Y_{\pi^{-1}(\sigma)}$.
3. The sender checks $U \in \{Y_1, \ldots, Y_n\}$ and sends $V = U^x$.
4. The sender proves that (g, h, U, V) in ZK-PoM that it is a DDH-tuple.
5. Note
$$V = U^x = Y^x_{\pi^{-1}(\sigma)} = X^{cx}_{\pi(\pi^{-1}(\sigma))} = (g^{s_\sigma} A_\sigma)^{cx}$$
so that $V^{1/c} = (g^{s_\sigma} A_\sigma)^x$, and hence $V^{1/c} h^{-s_\sigma} = A^x_\sigma$. The receiver now obtains M_σ via B_σ / A^x_σ.

The ZK-PoKs in the initialization phase are exactly the well-known Schnorr proof [19]. The ZK-PoM in transfer phases can be implemented using Chaum's technique [2].

Relation with Kurosawa-Nojima [10]. In the scheme of Kurosawa and Nojima [10], there are no steps 5-7 of shuffles in the initialization phase. Furthermore, their steps 2 and 3 in each transfer phase are as follows. First, $U = A^u$ for random value $u \in Z_q$ and some $A \in G$, both chosen by the receiver. The receiver is then required to persuade the sender that $A = A_\sigma$ for some $\sigma \in \{1, \ldots, n\}$. Obviously, the receiver cannot reveal A_σ (since otherwise, σ is revealed as well). Kurosawa and Nojima solved in [10] by mixing σ with other indexes in $\{1, \ldots, n\}$. Namely, they forced the receiver to prove in WI-PoK that he knows some $u \in Z_q$ satisfying
$$U = A^u_1 \vee \cdots \vee A^u_n.$$
The above WI-PoK, unfortunately, makes the communication cost of each transfer become $O(n)$.

In order to have $O(1)$ communication cost for each transfer phase, a possible method is to move the above WI-PoK to the initialization phase. Certainly, since the index σ of each transfer phase may be not chosen in advance, we move the WI-PoKs (each costs $O(n)$) corresponding to all possible n indexes, so that the

communication cost of the initialization phase becomes $O(n^2)$. Moving further, we mix the indexes by shuffling, and fortunately, by making use of existing results [16], the cost is better reduced to $O(n)$, which is asymptotically minimal for the initialization phase.

4.2 Security

We now have the following theorems ensuring the security of our adaptive OT protocol.

Theorem 1 *The above adaptive OT protocol is secure against sender corruption under the DDH assumption.*

Proof. For every real-world adversary $\mathcal{A}$ who corrupts the sender, we construct an ideal-world adversary $\mathcal{A}'$ such that the advantage $\mathbf{Adv}(\mathcal{Z})$ is negligible.

We will consider a sequence of games beginning from game G_0, which is the real world experiment, and proceed to the final game, which is the ideal world experiment as in Sec.2.3. For each integer i, let

$$\Pr(G_i) = \Pr(\mathcal{Z} = 1 \text{ in game } G_i),$$

and denote $\Pr(G_i) \approx \Pr(G_j)$ when the two values are negligibly close.

Game G_0: This is the real world experiment such that the sender is controlled by the adversary $\mathcal{A}$. By definition $\Pr(G_0) = \Pr(\mathcal{Z} = 1 \text{ in the real world})$.

Game G_1: This game is the same as the previous one except the following. In the initialization phase, the receiver extracts x from $\mathcal{A}$ by using the knowledge extractor of the ZK-PoK.

If it fails, then the protocol stops. Since the failure occurs with negligible probability, we have $\Pr(G_0) \approx \Pr(G_1)$.

Game G_2: This game is the same as game G_1 except that, in the initialization phase, the game uses the zero-knowledge simulators of the ZK-PoK at steps 5-7. Since the protocol at step 7 is computational zero-knowledge under the DDH assumption, and the others are perfect [4], we have $\Pr(G_1) \approx \Pr(G_2)$.

Game G_3: This game is the same as the previous one except that in the initialization phase, the receiver sends random $(Y_1, \ldots, Y_n) \in G^n$ to the sender.

We will prove $\Pr(G_3) \approx \Pr(G_2)$. Before that, let us state the following established result.

Fact 2 (Naor, Reingold [15]) *There exists a poly-time algorithm* Q *that, on input* (g, g^c, X^*, Y^*)*, outputs a random pair* $(X, Y) \in G^2$ *such that* (g, g^c, X, Y) *is a DDH tuple if and only if* (g, g^c, X^*, Y^*) *is.*

Lemma 3 $\Pr(G_3) \approx \Pr(G_2)$ *under the DDH assumption.*

Proof (of Lemma 3). By using $\mathcal{Z}$ and the corrupted sender $\mathcal{A}$, we construct a DDH distinguisher $\mathcal{D}$ as follows. On input $(g, C = g^c, X^*, Y^*)$, $\mathcal{D}$ first runs $\mathsf{Q}(g, C = g^c, X^*, Y^*)$ to generate the pairs $(X_1, Y_1), \ldots, (X_n, Y_n)$.

$\mathcal{D}$ next runs $\mathcal{Z}$ which sends $(M_1, \ldots, M_n)$ to $\mathcal{A}$ (the sender), and an index σ to the receiver. $\mathcal{A}$ and the receiver run the initialization phase until step 4. At step 5, $\mathcal{D}$ sends $C = g^c$ to $\mathcal{A}$, and runs the simulator of the ZK-PoK on c. At step 6, $\mathcal{D}$ sends the above $(X_1, \ldots, X_n)$ to $\mathcal{A}$, and runs the simulator of the ZK-PoK on $s_i (1 \leq i \leq n)$. At step 7, $\mathcal{D}$ sends the above $(Y_1, \ldots, Y_n)$ in random order to $\mathcal{A}$, and runs the zero-knowledge simulator of the shuffle protocol.

$\mathcal{A}$ and the receiver run the transfer phase as it is. Note that $\mathcal{D}$ can extract the secret key from $\mathcal{A}$, and hence extract M_i^* for all i (at the beginning), and $\mathcal{D}$ (playing the receiver) sends M_i^* to $\mathcal{Z}$ if necessary.

Finally, $\mathcal{A}$ sends $\mathcal{A}_{\text{out}}$ to $\mathcal{Z}$. The distinguisher $\mathcal{D}$ outputs what $\mathcal{Z}$ outputs.

One can see that if $\mathcal{D}$'s input $(g, C = g^c, X^*, Y^*)$ is a DDH tuple, then we are in game G_2; otherwise we are in game G_3, finishing the proof.

Game G_4: This game is the same as the previous one except the following. In each transfer phases, the receiver chooses U randomly and distinctly from the set $\{Y_1, \ldots, Y_n\}$. Since the view of $\mathcal{A}$ is unchanged, we have $\Pr(G_4) = \Pr(G_3)$.

Game G_5: This game is the ideal world experiment in which an ideal-world adversary $\mathcal{A}'$ uses $\mathcal{A}$ as a black-box as follows.

1. $\mathcal{A}'$ receives $(M_1, \ldots, M_n)$ from $\mathcal{Z}$, and sends $(M_1, \ldots, M_n)$ to $\mathcal{A}$.
2. $\mathcal{A}'$ runs **Game** G_4 with $\mathcal{A}$, where $\mathcal{A}'$ plays the role of the receiver. She can do this because σ (which is the secret of the receiver) is not used in **Game** G_4.
3. In the initialization phase, $\mathcal{A}'$ computes $M_i^* = B_i/(A_i)^x$ for all i by using x (which is extracted in **Game** G_1), and sends $(M_1^*, \ldots, M_n^*)$ to $\mathcal{F}_{\text{adapt}}$.
4. In each transfer phase, if $\mathcal{A}$ behaved in an acceptable way, then $\mathcal{A}'$ sends $b = 1$ to $\mathcal{F}_{\text{adapt}}$. Otherwise $\mathcal{A}'$ sends $b = 0$ to $\mathcal{F}_{\text{adapt}}$.
5. Suppose that $\mathcal{A}$ sends $\mathcal{A}_{\text{out}}$ to $\mathcal{Z}$ at the end of the game. Then $\mathcal{A}'$ sends $\mathcal{A}'_{\text{out}} = \mathcal{A}_{\text{out}}$ to $\mathcal{Z}$.

We have $\Pr(G_4) = \Pr(G_5)$, and by definition $\Pr(\mathcal{Z} = 1$ in the ideal world$) = \Pr(G_5)$. Summing up all above, we have $\mathbf{Adv}(\mathcal{Z}) = |\Pr(G_0) - \Pr(G_5)|$ is negligible as required. □

Theorem 4 *The above adaptive OT protocol is secure against receiver corruption under the DDH assumption.*

Proof. For every real-world adversary $\mathcal{A}$ who corrupts the receiver, we construct an ideal-world adversary $\mathcal{A}'$ such that the advantage of the environment $\mathbf{Adv}(\mathcal{Z})$ is negligible.

We again consider a sequence of games G_0, ..., G_6, where G_0 is the real world experiment of Sec.2.3, while G_6 is the ideal world experiment. Again, let $\Pr(G_i) = \Pr(\mathcal{Z} = 1$ in game $G_i)$.

Game G_0: In this game the receiver is controlled by the adversary $\mathcal{A}$, and by definition $\Pr(G_0) = \Pr(\mathcal{Z} = 1$ in the real world$)$.

Game G_1: This game is the same as game G_0 except the following. In the initialization phase, the sender extracts c and s_i by using the extractors of the ZK-PoK.

If it fails, then the protocol fails. Since this failure occurs with negligible probability, we have $\Pr(G_1) \approx \Pr(G_0)$.

Game G_2: This game is the same as the previous one except the following. First the sender extracts π by comparing $(X_1^c, \ldots, X_n^c)$ and $(Y_1, \ldots, Y_n)$. Next in each transfer phase, the sender extracts the index σ that $\mathcal{A}$ really used as follows.

$\mathcal{A}$ sends U such that $U \in \{Y_1, \ldots, Y_n\}$. The sender searches the index ρ satisfying $U = Y_\rho$. Recall $U = Y_{\pi^{-1}(\sigma)}$, so $\pi^{-1}(\sigma) = \rho$, and hence $\sigma = \pi(\rho)$. Thus the sender can extract σ that $\mathcal{A}$ really used.

Since the change is syntactic, we have $\Pr(G_2) = \Pr(G_1)$.

Game G_3: This game is the same as the previous one except the following. In each transfer phase, the sender computes V as $(B_\sigma M_\sigma^{-1} h^{s_\sigma})^c$. Since the change is syntactic, we have $\Pr(G_3) = \Pr(G_2)$.

Game G_4: This game is the same as the previous one except the following. In each transfer phase, instead of running the ZK-PoM which proves that (g, h, U, V) is a DDH-tuple, the zero-knowledge simulator of the ZK-PoM is run so that $\Pr(G_4) \approx \Pr(G_3)$.

Game G_5: This game is the same as the previous one except the following. In the initialization phase, each B_i is a random element of G. It is easy to see that $\Pr(G_5) \approx \Pr(G_4)$ under the DDH assumption.

Game G_6: This game is the ideal world experiment in which an ideal-world adversary $\mathcal{A}'$ uses $\mathcal{A}$ as a black-box as follows.

1. $\mathcal{A}'$ runs **Game** G_5 with $\mathcal{A}$, where $\mathcal{A}'$ plays the role of the sender.
2. In each transfer phase, $\mathcal{A}'$ sends σ which is extracted as in **Game** G_2 to $\mathcal{F}_{\text{adapt}}$, and obtains M_σ. $\mathcal{A}'$ then computes V as in **Game** G_3.
3. Suppose that $\mathcal{A}$ sends $\mathcal{A}_{\text{out}}$ to $\mathcal{Z}$ at the end of the game. Then $\mathcal{A}'$ sends $\mathcal{A}'_{\text{out}} = \mathcal{A}_{\text{out}}$ to $\mathcal{Z}$.

We have by definition $\Pr(G_6) = \Pr(\mathcal{Z} = 1$ in the ideal world$)$. Summing up all above, we have $\mathbf{Adv}(\mathcal{Z}) = |\Pr(G_0) - \Pr(G_6)|$ is negligible as required. □

References

1. J. Camenisch, G. Neven, and A. Shelat. Simulatable adaptive oblivious transfer. In M. Naor, editor, *EUROCRYPT*, volume 4515 of *Lecture Notes in Computer Science*, pages 573–590. Springer, 2007.
2. D. Chaum. Zero-knowledge undeniable signatures. In *EUROCRYPT*, pages 458–464, 1990.

3. C.-K. Chu and W.-G. Tzeng. Efficient 1-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries. In S. Vaudenay, editor, *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 2005.
4. R. Cramer, I. Damgård, and P. D. MacKenzie. Efficient zero-knowledge proofs of knowledge without intractability assumptions. In H. Imai and Y. Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 354–373. Springer, 2000.
5. O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for np. *J. Cryptology*, 9 (3):167–190, 1996.
6. M. Green and S. Hohenberger. Universally composable adaptive oblivious transfer. In J. Pieprzyk, editor, *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 179–197. Springer, 2008.
7. M. Green and S. Hohenberger. Practical adaptive oblivious transfer from a simple assumption. Cryptology ePrint Archive, Report 2010/109, 2010. `http://eprint.iacr.org/`.
8. S. Jarecki and X. Liu. Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 577–594. Springer, 2009.
9. Y. T. Kalai. Smooth projective hashing and two-message oblivious transfer. In R. Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 78–95. Springer, 2005.
10. K. Kurosawa and R. Nojima. Simple adaptive oblivious transfer without random oracle. In M. Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 334–346. Springer, 2009.
11. M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *STOC*, pages 245–254, 1999.
12. M. Naor and B. Pinkas. Oblivious transfer with adaptive queries. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 573–590. Springer, 1999.
13. M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In *SODA*, pages 448–457, 2001.
14. M. Naor and B. Pinkas. Computationally secure oblivious transfer. *J. Cryptology*, 18(1):1–35, 2005.
15. M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004.
16. C. A. Neff. A verifiable secret shuffle and its application to e-voting. In *ACM Conference on Computer and Communications Security*, pages 116–125, 2001.
17. C. A. Neff. Shuffles of ElGamal pairs, 2004. Available at `http://people.csail.mit.edu/rivest/voting/`.
18. W. Ogata and K. Kurosawa. Oblivious keyword search. *J. Complexity*, 20(2-3):356–371, 2004. Also available at `http://eprint.iacr.org/2002/182`.
19. C.-P. Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.