Improvement of One Adaptive Oblivious Transfer Scheme

Zhengjun Cao [1], Lihua Liu [2,*]

Abstract

In 2011, the authors [8] presented an adaptive oblivious transfer (OT) scheme based on Decisional 3-Party Diffie-Hellman (3DDH) assumption. The encryption used in the scheme is a combination of the Boneh-Boyen IBE scheme and a variation of the Hohenberger-Waters signature. The scheme is somewhat inefficient since it combines the two underlying schemes in a simple way. In this paper, we present an improvement of the OT scheme and show its security under 3DDH assumption. The proposed skills are helpful for designing and analyzing other cryptographic schemes.

Keywords. adaptive oblivious transfer; 3-Party Diffie-Hellman assumption; redundant system parameters.

1 Introduction

Oblivious Transfer, introduced by Rabin [16], is of fundamental importance in multi-party computation [9, 18]. In an adaptive oblivious transfer protocol, a sender commits to a database of messages and then repeatedly interacts with a receiver in such a way that the receiver obtains one message per interaction of his choice (and nothing more) while the sender learns nothing about any of the choices. For the related works, we refer to [3,5-8, 11-14,17].

In 2011, the authors [8] presented an adaptive oblivious transfer scheme based on Decisional 3-Party Diffie-Hellman assumption which says that given (g, g^a, g^b, g^c, Q) where g generates a bilinear group of prime order p and a, b, c are selected randomly from $\mathbb{Z}_p$, it is hard to decide if $Q = g^{abc}$. In the scheme, the sender commits to a database of n messages by publishing an encryption of each message and a signature on each encryption. Then, each transfer phase can be executed in time independent of n as the receiver blinds one of the encryptions and proves knowledge of the blinding factors and a signature on this encryption, after which the sender helps the receiver decrypt the chosen ciphertext.

[01] Department of Mathematics, Shanghai University, Shanghai, China.
[2] Department of Mathematics, Shanghai Maritime University, China. liulh@shmtu.edu.cn

The encryption used in the scheme is a combination of the Boneh-Boyen IBE scheme [1] and a variation of the Hohenberger-Waters signature [10]. However, it combines the two underlying schemes in a simple way. Concretely, there are two drawbacks: (1) It sets the secret key as (a, b), where a is used only for decryption and b is used only for signing, separately. But we know it is usual that a single secret key a can be used simultaneously for both signing and decryption. (2) For random $r, s, t \in \mathbb{Z}_p$, it expresses the ciphertext as

$$C = \Big(g^r,\ (g_1^j h)^r,\ M \cdot e(g_1, g_2)^r,\ g^t,\ (u^r v^s d)^b (g_3^j h)^t,\ u^r,\ s\Big)$$

where $p, g, e, g_1, g_2, g_3, g_4, u, v, d, h$ are included in public parameters. The session key s is *directly exposed*. That means the corresponding parameter v could be reasonably removed.

In this paper, we present an improvement of the adaptive OT scheme [8] and show its security under 3DDH assumption. We also correct some typos in the original scheme. The analysis skills presented in the paper is novel. We think it is helpful for optimizing some cryptographic schemes.

2 Preliminaries

Let *BMsetup* be an algorithm that, on input 1^κ, outputs the parameters for a bilinear mapping as $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$, where g generates $\mathbb{G}$, the groups $\mathbb{G}$ and $\mathbb{G}_T$ have prime order p, and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. It is both: (*bilinear*) for all $g \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, $e(g^a,\ g^b) = e(g,\ g)^{ab}$; and (*non-degenerate*) if g generates $\mathbb{G}$, then $e(g,\ g) \neq 1$.

Assumption 2.1. (Decisional 3-Party Diffie-Hellman (3DDH) [2]) *Let g generate a group $\mathbb{G}$ of prime order $p \in \Theta(2^\lambda)$. For all p.p.t. adversaries $\mathcal{A}$, the following probability is 1/2 plus an amount negligible in λ:*

$$\Pr\left[g, z_0 \leftarrow \mathbb{G}; a, b, c \leftarrow \mathbb{Z}_p;\ z_1 \leftarrow g^{abc}; d \leftarrow \{0,1\}; d' \leftarrow \mathcal{A}(g, g^a, g^b, g^c, z_d) : d = d'\right].$$

We use the notation of Camenisch and Stadler [4] for the proofs of knowledge. For instance, $ZKPoK\{(x, h) : y = g^x \wedge H = e(y, h) \wedge (1 \leq x \leq n)\}$ denotes a zero-knowledge proof of knowledge of an integer x and a group element $h \in \mathbb{G}$ such that $y = g^x$ and $H = e(y, h)$ holds and $1 \leq x \leq n$. All values not enclosed in ()'s are assumed to be known to the verifier.

3 Definition of adaptive k-out-of-N oblivious transfer ($\mathbf{OT}_{k\times 1}^N$)

The definition can be found in Ref.[8]. For completeness, we now relate it as follows. An adaptive oblivious transfer scheme is a tuple of algorithms $(\mathsf{S_I}, \mathsf{R_I}, \mathsf{S_T}, \mathsf{R_T})$. During the initialization phase, the Sender and the Receiver conduct an interactive protocol, where the Sender runs $\mathsf{S_I}(M_1, \cdots, M_N)$ to obtain state value S_0, and the Receiver runs $\mathsf{R_I}()$ to obtain state value R_0. Next, for $1 \le i \le k$, the i^{th} transfer proceeds as follows: the Sender runs $\mathsf{S_T}(S_{i-1})$ to obtain state value S_i, and the Receiver runs $\mathsf{R_T}(R_{i-1}, \sigma_i)$ where $1 \le \sigma_i \le N$ is the index of the message to be received. The receiver obtains state information R_i and the message M'_{σ_i} or $\perp$ indicating failure. To define the Sender and Receiver security, we need the following experiments.

Real experiment. In experiment $\mathbf{Real}_{\hat{\mathsf{S}},\hat{\mathsf{R}}}(N, k, M_1, \cdots, M_N, \Sigma)$, the possibly cheating sender $\hat{\mathsf{S}}$ is given messages $(M_1, \cdots, M_N)$ as input and interacts with the possibly cheating receiver $\hat{\mathsf{R}}(\Sigma)$, where Σ is a selection algorithm that on input the full collection of messages thus far received, outputs the index σ_i of the next message to be queried. At the beginning of the experiment, both $\hat{\mathsf{S}}$ and $\hat{\mathsf{R}}$ output initial states (S_0, R_0). In the transfer phase, for $1 \le i \le k$ the sender computes $S_i \leftarrow \hat{\mathsf{S}}(S_{i-1})$, and the receiver computes $(R_i, M'_i) \leftarrow \hat{\mathsf{R}}(R_{i-1})$, where M'_i may or may not be equal to M_i. At the end of the k-th transfer the output of the experiment is (S_k, R_k).

Ideal experiment. In experiment $\mathbf{Ideal}_{\hat{\mathsf{S}}',\hat{\mathsf{R}}'}(N, k, M_1, \cdots, M_N, \Sigma)$ the possibly cheating sender algorithm $\hat{\mathsf{S}}'$ generates messages $(M_1^*, \cdots, M_N^*)$ and transmits them to a trusted party T. In the i-th round $\hat{\mathsf{S}}'$ sends a bit b_i to T; the possibly cheating receiver $\hat{\mathsf{R}}'(\Sigma)$ transmits σ_i^* to T. If $b_i = 1$ and $\sigma_i^* \in \{1, \cdots, N\}$ then T hands $M^*_{\sigma_i^*}$ to $\hat{\mathsf{R}}'$. If $b_i = 0$ then T hands $\perp$ to $\hat{\mathsf{R}}'$. After the k-th transfer the output of the experiment is (S_k, R_k).

Sender Security. An $\mathrm{OT}^N_{k\times 1}$ provides Sender security if for every real-world p.p.t. receiver $\hat{\mathsf{R}}$ there exists a p.p.t. ideal-world receiver $\hat{\mathsf{R}}'$ such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \cdots, M_N)$, Σ, and every p.p.t. distinguisher: $\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}(N, k, M_1, \cdots, M_N, \Sigma) \overset{c}{\approx} \mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}(N, k, M_1, \cdots, M_N, \Sigma)$, where $\ell(\cdot)$ is a polynomially-bounded function.

Receiver Security. An $\mathrm{OT}^N_{k\times 1}$ provides Receiver security if for every real-world p.p.t. sender $\hat{\mathsf{S}}$ there exists a p.p.t. ideal-world sender $\hat{\mathsf{S}}'$ such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \cdots, M_N)$, Σ, and every p.p.t. distinguisher: $\mathbf{Real}_{\hat{\mathsf{S}},\mathsf{R}}(N, k, M_1, \cdots, M_N, \Sigma) \overset{c}{\approx} \mathbf{Ideal}_{\hat{\mathsf{S}}',\mathsf{R}'}(N, k, M_1, \cdots, M_N, \Sigma)$.

4 Review and analysis of one adaptive OT scheme

4.1 Review

This protocol follows the assisted (or blind) decryption paradigm [3, 7, 11]. The Sender begins the OT protocol by encrypting each message in the database and publishing these values to the Receiver. The Receiver then checks that each ciphertext is well-formed. See the following Table 1 for details.

Table 1: The Green-Hohenberger OT scheme

$\mathsf{S_I}(M_1, \cdots, M_N)$	$\mathsf{R_I}()$
1. Select $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow$ BMsetup (1^κ) and $a, b \leftarrow \mathbb{Z}_p$, choose $g_2, g_3, h, u, v, d \leftarrow \mathbb{G}$ and set $g_1 \leftarrow g^a, g_4 \leftarrow g^b$. Let $pk \leftarrow (\gamma, g_1, g_2, g_3, g_4, h, u, v, d)$, $sk \leftarrow (a, b)$. 2. For $j = 1$ to N, select $r_j, s_j, t_j \leftarrow \mathbb{Z}_p$ and set: $C_j \leftarrow [g^{r_j}, (g_1^j h)^{r_j}, M_j e(g_1, g_2)^{r_j},$ $g^{t_j}, (u^{r_j} v^{s_j} d)^b (g_3^j h)^{t_j}, u^{r_j}, s_j]$ 3. Send $(pk, C_1, \cdots, C_N)$ to Receiver. 4. Conduct $ZKPoK\{(a) : g_1 = g^a\}$.	
	5. Verify pk and the proof. Check for $j = 1$ to N: VerifyCiphertext (pk, C_j, j)=1. If any check fails, output $\perp$.
Output $S_0 = (pk, sk)$.	Output $R_0 = (pk, C_1, \cdots, C_N)$.
$\mathsf{S_T}(S_{i-1})$	$\mathsf{R_T}(R_{i-1}, \sigma_i)$ 1. Parse C_{σ_i} as $(c_1, \cdots, c_7)$, select $\underline{x, y \leftarrow \mathbb{Z}_p}$ and compute $v_1 = g^x c_1$. 2. Send v_1 to Sender, and conduct: $WIPoK\{(\sigma_i, x, c_2, c_4, c_5, c_6, c_7) :$ $e(v_1/g^x, (g_1^{\sigma_i} h)) = e(c_2, g) \wedge$ $e(c_6, g) = e(v_1/g^x, u) \wedge$ $e(c_5, g) = e(c_6 v^{c_7} d, g_4) e(c_4, g_3^{\sigma_i} h)\}$
3. Set $R = e(v_1, g_2^a)$. 4. Send R to Receiver and conduct: $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.	
	5. If the proof does not verify, output $\perp$. Else output $M'_{\sigma_i} = \frac{c_3 \cdot e(g_1, g_2)^x}{R}$.
Output $S_i = S_{i-1}$.	Output $R_i = (R_{i-1}, M'_{\sigma_i})$

Ciphertext Structure. The Sender's public parameters pk include $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$ and generators $(g_1, g_2, h, g_3, g_4, u, v, d) \in \mathbb{G}^8$. For message $M \in \mathbb{G}_T$, identity $j \in \mathbb{Z}_p$, and random values $r, s, t \in \mathbb{Z}_p$, the ciphertext is expressed as: $C = \left(g^r, (g_1^j h)^r, M \cdot e(g_1, g_2)^r, g^t, (u^r v^s d)^b (g_3^j h)^t, u^r, s\right)$. Given only pk, j, the VerifyCiphertext function validates that the ciphertext has this structure.

VerifyCiphertext(pk, C, j). Parse C as $(c_1, \cdots, c_7)$ and pk to obtain $g, g_1, h, g_3, g_4, u, v, d$. This routine outputs 1 if and only if the following equalities hold:

$$e(g_1^j h, c_1) = e(g, c_2) \;\wedge\; e(g, c_6) = e(c_1, u) \wedge$$
$$e(g, c_5) = e(g_4, c_6 v^{c_7} d) e(c_4, g_3^j h)$$

4.2 Drawbacks

The encryption used in the scheme is a combination of the Boneh-Boyen IBE scheme [1] and a variation of the Hohenberger-Waters signature [10]. It combines the two base schemes in a simple way. Concretely, there are three drawbacks:

(I) It sets the secret key as (a, b), where a is used only for decryption and b is used only for signing, separately. But is is usual that a single secret key a can be simultaneously used for both signing and decryption. We will set $b = a$ and show that the setting does not endanger its security. That means the generator g_4 could be removed.

(II) For random $r, s, t \in \mathbb{Z}_p$, it expresses the ciphertext as

$$C = \Big(g^r,\, (g_1^j h)^r,\, M \cdot e(g_1, g_2)^r,\, g^t,\, (u^r v^s d)^b (g_3^j h)^t,\, u^r,\, s\Big) \tag{1}$$

Notice that the session key s is *directly exposed*. That means the generator v could be removed, too. The redundant setting is due to that the authors follow the Hohenberger-Waters signature based on RSA assumption (see Section 3 in Ref.[10]), which does require a chameleon hash function. We would like to stress that the structure $u^M v^s$ in a bilinear group $\mathbb{G}$ has no the special property of a chameleon hash function because one can not find s' satisfying $u^M v^s = u^{M'} v^{s'}$, given M, M' and s, where u, v are two random elements of $\mathbb{G}$. The authors misapplied the structure.

(III) The generator g_2 is used only for the blind decryption and the generator g_3 is used only for the VerifyCiphertext. For simplicity, we could explicitly set that $g_3 = g_2$. That is to say, the generator g_3 might be redundant. By the way, the generator d is required necessarily for the Hohenberger-Waters signature based on CDH assumption [10]. The generator h facilitates the security proof of the Hohenberger-Waters signature. If d is removed, then we have the following attack. Given a valid ciphertext

$$C = (c_1, \cdots, c_7) = \Big(g^r, (g_1^j h)^r, M \cdot e(g_1, g_2)^r, g^t, (u^r v^s)^b (g_3^j h)^t, u^r, s\Big) \tag{2}$$

an adversary can take a random $\theta \in \mathbb{Z}_p$ and compute

$$\hat{C} = (\hat{c}_1, \cdots, \hat{c}_7) = \left(g^{r\theta}, (g_1^j h)^{r\theta}, M^\theta \cdot e(g_1, g_2)^{r\theta}, g^{t\theta}, \Big((u^r v^s)^b (g_3^j h)^t\Big)^\theta, u^{r\theta}, s\theta\right) \tag{3}$$

The ciphertext $\hat{C}$ is valid because

$$e(g_1^j h, \hat{c}_1) = e(g, \hat{c}_2) \;\wedge\; e(g, \hat{c}_6) = e(\hat{c}_1, u) \wedge$$
$$e(g, \hat{c}_5) = e(g_4, \hat{c}_6 v^{\hat{c}_7}) e(\hat{c}_4, g_3^j h)$$

Remark 4.1. The random $y \in \mathbb{Z}_p$ chosen by the receiver is not used at all. This is a typo.

5 An improvement and its security proof

5.1 The improvement

The improvement is obtained by removing the redundant generators g_3, g_4, v. See the table 2 for details.

Table 2: The improvement

$\mathsf{S_I}(M_1, \cdots, M_N)$	$\mathsf{R_I}()$
1. Select $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow$ BMsetup (1^κ) and $a \leftarrow \mathbb{Z}_p$, choose $g_2, h, u, d \leftarrow \mathbb{G}$ and set $g_1 \leftarrow g^a$. Let $pk \leftarrow (\gamma, g_1, g_2, h, u, d)$, $sk \leftarrow a$.	
2. For $j = 1$ to N, select $r_j, t_j \leftarrow \mathbb{Z}_p$ and set: $C_j \leftarrow [g^{r_j}, (g_1^j h)^{r_j}, M_j\, e(g_1, g_2)^{r_j}, g^{t_j}, (u^{r_j} d)^a (g_2^j h)^{t_j}, u^{r_j}]$	
3. Send $(pk, C_1, \cdots, C_N)$ to Receiver.	
4. Conduct $ZKPoK\{(a) : g_1 = g^a\}$.	
	5. Verify pk and the proof. Check for $j = 1$ to N: $\mathsf{VerifyCiphertext}$ (pk, C_j, j)=1. If any check fails, output $\perp$.
Output $S_0 = (pk, sk)$.	Output $R_0 = (pk, C_1, \cdots, C_N)$.
$\mathsf{S_T}(S_{i-1})$	$\mathsf{R_T}(R_{i-1}, \sigma_i)$
	1. Parse C_{σ_i} as $(c_1, \cdots, c_6)$, select $x \leftarrow \mathbb{Z}_p$ and compute $v_1 = g^x c_1$.
	2. Send v_1 to Sender, and conduct: $WIPoK\{(\sigma_i, x, c_2, c_4, c_5, c_6) : e(v_1/g^x, (g_1^{\sigma_i} h)) = e(c_2, g) \wedge e(c_6, g) = e(v_1/g^x, u) \wedge e(c_5, g) = e(c_6 d, g_1) e(c_4, g_2^{\sigma_i} h)\}$
3. Set $R = e(v_1, g_2^a)$.	
4. Send R to Receiver and conduct: $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.	
	5. If the proof does not verify, output $\perp$. Else output $M'_{\sigma_i} = \frac{c_3 \cdot e(g_1, g_2)^x}{R}$.
Output $S_i = S_{i-1}$.	Output $R_i = (R_{i-1}, M'_{\sigma_i})$

Ciphertext Structure. The Sender's public parameters pk include $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$ and generators $(g_1, g_2, h, u, d) \in \mathbb{G}^5$. For message $M \in \mathbb{G}_T$, identity $j \in \mathbb{Z}_p$, and random values $r, t \in \mathbb{Z}_p$, the ciphertext is expressed as: $C = \left(g^r, (g_1^j h)^r, M \cdot e(g_1, g_2)^r, g^t, (u^r d)^a (g_2^j h)^t, u^r\right)$. Given only pk, j, the $\mathsf{VerifyCiphertext}$ function validates that the ciphertext has this structure.

$\mathsf{VerifyCiphertext}(pk, C, j)$. Parse C as $(c_1, \cdots, c_6)$ and pk to obtain g, g_1, g_2, h, u, d. This routine outputs 1 if and only if the following equalities hold:

$$e(g_1^j h, c_1) = e(g, c_2) \ \wedge \ e(g, c_6) = e(c_1, u) \wedge$$
$$e(g, c_5) = e(g_1, c_6 d) e(c_4, g_2^j h)$$

Correctness.

$$e(g_1^j h, c_1) = e(g_1^j h, g^{r_j}) = e((g_1^j h)^{r_j}, g) = e(g, c_2)$$
$$e(g, c_6) = e(g, u^{r_j}) = e(g^{r_j}, u) = e(c_1, u)$$
$$e(g, c_5) = e\left(g, (u^{r_j} d)^a (g_2^j h)^{t_j}\right) = e\left(g, (u^{r_j} d)^a\right) e\left(g, (g_2^j h)^{t_j}\right) = e(g_1, c_6 d) e(c_4, g_2^j h)$$
$$\frac{c_3 \cdot e(g_1, g_2)^x}{R} = \frac{M_j \, e(g_1, g_2)^{r_j} \cdot e(g_1, g_2)^x}{e(g^x c_1, g_2^a)} = \frac{M_j \, e(g_1, g_2)^{r_j} \cdot e(g_1, g_2)^x}{e(g^x, g_2^a) e(g^{r_j}, g_2^a)} = M_j$$

5.2 Security proof

The improvement is sender-secure and receiver-secure in the full simulation model under 3DDH assumption. The security proof is very like that of the original scheme [8]. For completeness, we now describe it as follows.

Sender security. Given a (possibly cheating) real-world receiver $\hat{\mathsf{R}}$, we show how to construct an ideal-world receiver $\hat{\mathsf{R}}'$ such that all p.p.t. distinguishers have at most negligible advantage in distinguishing the distribution of an honest real-world sender S interacting with $\hat{\mathsf{R}}$ ($\text{Real}_{\mathsf{S}, \hat{\mathsf{R}}}$) from that of $\hat{\mathsf{R}}'$ interacting with the honest ideal-world sender S' ($\text{Ideal}_{\mathsf{S}', \hat{\mathsf{R}}'}$).

1. To begin, $\hat{\mathsf{R}}'$ selects a random collection of messages $\bar{M}_1, \cdots, \bar{M}_N \leftarrow \mathbb{G}_T$ and follows the $\mathsf{S_I}$ algorithm with these as input up to the point where it obtains $(pk, C_1, \cdots, C_N)$.

2. It sends $(pk, C_1, \cdots, C_N)$ to $\hat{\mathsf{R}}$ and then simulates the interactive proof $ZKPoK\{(a) : g_1 = g^a\}$. (Even though $\hat{\mathsf{R}}'$ knows $sk = a$, it ignores this value and simulate this proof step.)

3. For each of k transfers initiated by $\hat{\mathsf{R}}$,

(a) $\hat{\mathsf{R}}'$ verifies the received WIPoK and uses the knowledge extractor E_2 to obtain the values $\sigma_i, x, c_1, c_2, c_3, c_4$ from it. $\hat{\mathsf{R}}'$ aborts and outputs error when E_2 fails.

(b) When $\sigma_i \in [1, N]$, $\hat{\mathsf{R}}'$ queries the trusted party T to obtainM_{σ_i}, parses C_{σ_i} as $(c_1, \cdots, c_6)$ and responds with $R = \frac{c_3 \, e(g_1, g_2)^x}{M_{\sigma_i}}$ (if T returns $\perp$, $\hat{\mathsf{R}}'$ aborts the transfer). When $\sigma_i \notin [1, N]$, $\hat{\mathsf{R}}'$

follows the normal protocol. In both cases, $\hat{\mathsf{R}}'$ simulates $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.

4. $\hat{\mathsf{R}}'$ uses $\hat{\mathsf{R}}$'s output as its own.

Theorem 5.1 *Let ϵ_{ZK} be the maximum advantage with which any p.p.t. algorithm distinguishes a simulated ZKPoK, and ϵ_{Ext} be the maximum probability that the extractor E_2 fails (with ϵ_{ZK} and ϵ_{Ext} both negligible in κ). If all p.p.t. algorithms have negligible advantage $\leq \epsilon$ at solving the 3DDH problem, then:*

$$\Pr\left[D(\text{Real}_{\mathsf{S},\hat{\mathsf{R}}}(N, k, M_1, \cdots, M_N, \Sigma)) = 1\right] - \Pr\left[D(\text{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}(N, k, M_1, \cdots, M_N, \Sigma)) = 1\right] \leq (k+1)\epsilon_{ZK} + k\epsilon_{Ext} + N\epsilon\left(1 + \frac{p}{p-1}\right).$$

Proof. We first define the following games:

Game 0. The real-world experiment conducted between S and $\hat{\mathsf{R}}$ ($\text{Real}_{\mathsf{S},\hat{\mathsf{R}}}$).

Game 1. This game modifies **Game 0** as follows: (1) each of S's ZKPoK executions is replaced with a simulated proof of the same statement, and (2) the knowledge extractor E_2 is used to obtain the values[1] $(\sigma_i, x, \bar{c}_4, \bar{c}_5, \bar{c}_6)$ from each of $\hat{\mathsf{R}}$'s transfer queries. Whenever the extractor fails, S terminates the experiment and outputs the distinguished symbol error.

Game 2. This game modifies **Game 1** such that, whenever the extracted value $\sigma_i \in [1, N]$, S's response R is computed using the following approach: parse $C_{\sigma_i} = (c_1, \cdots, c_6)$ and set $R = \frac{c_3\, e(g_1, g_2)^x}{M_{\sigma_i}}$. When $\sigma_i \notin [1, N]$, the response is computed using the normal protocol.

Game 3. This game modifies **Game 2** by replacing the input to S_I with a dummy vector of random messages $\bar{M}_1, \cdots, \bar{M}_N \in \mathbb{G}_T$. However when S computes a response value using the technique of **Game 2**, the response is based on the original message vector $M_1, \cdots, M_N$. We claim that the distribution of this game is equivalent to that of $\text{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}$.

For notational convenience, define:

$$\text{Adv[Game i]} = \Pr[D(\text{Game i}) = 1] - \Pr[D(\text{Game 0}) = 1].$$

[1] There is a typo in the original argument. It says that "the knowledge extractor E_2 is used to obtain the values $(\sigma_i, x, y, z, \bar{c}_4, \bar{c}_5, \bar{c}_6, \bar{c}_7)$ from each of $\hat{\mathsf{R}}$'s transfer queries". We should stress that both the values y, z are not used at all.

By the following Lemmas, we then obtain $\text{Adv[Game 3]} \leq (k+1)\epsilon_{ZK} + k\epsilon_{Ext} + N\epsilon(1+\frac{p}{p-1})$. $\square$

Lemma 5.2 *If all p.p.t. algorithms D distinguish a simulated ZKPoK with advantage at most ϵ_{ZK} and the extractor E_2 fails with probability at most ϵ_{Ext}, then* $\text{Adv[Game 1]} \leq (k+1)\epsilon_{ZK} + k\epsilon_{Ext}$.

Proof. See the proof of Lemma A.1 in Ref.[8]. $\square$

Lemma 5.3 *If no p.p.t. algorithm has advantage $> \epsilon$ in solving the 3DDH problem, then*

$$\text{Adv[Game 2]} - \text{Adv[Game 1]} \leq \frac{Np}{p-1} \cdot \epsilon$$

Proof. For every query where $\sigma_i \notin [1, N]$, S calculates the response R as in the normal protocol, and thus the distribution of R is identical to **Game 1**. Thus we need only consider queries where $\sigma_i \in [1, N]$.

Given a transfer request containing v_1, let us implicitly define $g^{r'} = v_1/g^x$ for some $r' \in \mathbb{Z}_p$. Express the σ_i-th ciphertext in the database as $C_{\sigma_i} = (c_1, \cdots, c_6)$. If $g^{r'} = c_1$ then the computed response R will have the same distribution as in the normal protocol. To show this, let $c_1 = g^{r_{\sigma_i}}$ for some $r_{\sigma_i} \in \mathbb{Z}_p$ and $c_3/M_{\sigma_i} = e(g_1, g_2)^{r_{\sigma_i}}$. We can now write the normal calculation of R as:

$$R = e(c_1 g^x, g_2^a) = e(g^{r_{\sigma_i}} g^x, g_2^a) = e(g_1, g_2)^{r_{\sigma_i}} e(g_1, g_2)^x = \frac{c_3\, e(g_1, g_2)^x}{M_{\sigma_i}}$$

It remains only to consider the case where $g^{r'} \neq c_1$. We will refer to this as a *forged query* and argue that $\hat{\mathsf{R}}$ cannot issue such a query except with negligible probability under the 3DDH assumption in $\mathbb{G}$. Specifically, if $\hat{\mathsf{R}}$ submits a forged query with non-negligible probability, then we can construct a solver $\mathcal{B}$ for 3DDH that succeeds with non-negligible advantage.

We now describe the solver $\mathcal{B}$. $\mathcal{B}$ takes as input a 3DDH tuple $(g, g^\tau, g^\psi, g^\omega, Z)$, where $Z = g^{\tau\psi\omega}$ or is random, and each value τ, φ, ω was chosen at random from $\mathbb{Z}_p$. It will simulate S's interaction with $\hat{\mathsf{R}}$ via the following simulation.

Simulation Setup. $\mathcal{B}$ first picks $j^* \leftarrow [1, N]$ and[2] $y_d, x_d, x_h, x_z \leftarrow \mathbb{Z}_p$. It sets $u = g^\psi$, $d = g^{-\psi x_d} g^{y_d}$, $h = g^{-\psi j^*} g^{x_h}$, $g_2 = g^\psi g^{x_z}$, $g_1 = g^\tau$. Thus, we implicitly have $a = \tau$. The remaining components of pk are chosen as in the real protocol.

For $j = 1$ to N, $\mathcal{B}$ generates each correctly-distributed ciphertext $C_j = (c_1, \cdots, c_6)$ as follows:

The simulation for $j = j^*$. Pick $t_j \leftarrow \mathbb{Z}_p$ and set the ciphertext as:

$$(c_1, \cdots, c_6) = \left(g^{x_d}, (g_1^j h)^{x_d}, M \cdot e(g_1, g_2)^{x_d}, g^{t_j}, (g^\tau)^{y_d} (g_2^j h)^{t_j}, u^{x_d}\right)$$

[2]There is a typo in the original argument. It says that "$\mathcal{B}$ first picks $j^* \leftarrow [1, N]$ and $a, y_v, y_d, x_v, x_d, x_h, x_z, r_j, t_j \leftarrow \mathbb{Z}_p$". Clearly, the secret key a for decryption is not known to the solver $\mathcal{B}$. Besides, it is not necessary for $\mathcal{B}$ to pick r_j, t_j in the Setup because they are not used at all in the phase.

The ciphertext is well-formed because:

$$e(g_1^j h, c_1) = e(g_1^j h, g^{x_d}) = e((g_1^j h)^{x_d}, g) = e(g, c_2)$$
$$e(g, c_6) = e(g, u^{x_d}) = e(g^{x_d}, u) = e(c_1, u)$$
$$e(g, c_5) = e\left(g, (g^\tau)^{y_d}(g_2^j h)^{t_j}\right) = e\left(g, (u^{x_d} d)^\tau\right) e\left(g, (g_2^j h)^{t_j}\right) = e(g_1, c_6 d) e(c_4, g_2^j h)$$

The simulation for $j \neq j^*$. Pick $r_j, t'_j \leftarrow \mathbb{Z}_p$. Set $Y = g^{t'_j}/(g^\tau)^{(r_j - x_d)/(j-j^*)}$ and the ciphertext as:

$$(c_1, \cdots, c_6) = \left(g^{r_j}, (g_1^j h)^{r_j}, M \cdot e(g_1, g_2)^{r_j}, Y, (g^\tau)^{y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j-j^*)}, u^{r_j}\right)$$

Let us define $Y = g^{t_j}$ and thus implicitly $t_j = t'_j - \tau(r_j - x_d)/(j - j^*)$, which is randomly distributed in $\mathbb{Z}_p$. Just by inspection, it's clear that all of the elements except c_5 are correctly distributed. Thus it remains to show that:

$$(g^\tau)^{y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j-j^*)} = (u^{r_j} d)^\tau (g_2^j h)^{t_j}$$

In fact, we have:

$$\begin{aligned}
c_5 &= (g^\tau)^{y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j-j^*)} \\
&= (g^\tau)^{y_d} \cdot (g^{t_j})^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j-j^*)} \\
&= (g^{\tau\psi})^{r_j - x_d} (g^\tau)^{y_d} \cdot (g^{t_j})^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j-j^*)} (g^{-\tau\psi})^{r_j - x_d} \\
&= (g^{\psi(r_j - x_d)})^\tau (g^{y_d})^\tau \cdot (g^{x_z j + x_h})^{t_j} \cdot (g^\psi)^{t'_j (j-j^*)} (g^{-\tau\psi})^{r_j - x_d} \\
&= ((g^{\psi r_j})(g^{-\psi x_d + y_d}))^\tau \cdot (g^{x_z j + x_h})^{t_j} \cdot (g^\psi)^{t'_j (j-j^*)} (g^{-\tau\psi})^{r_j - x_d} \\
&= (u^{r_j} d)^\tau \cdot (g^{x_z j + x_h})^{t_j} \cdot (g^\psi)^{t'_j (j-j^*)} (g^{-\tau\psi})^{r_j - x_d} \\
&= (u^{r_j} d)^\tau \cdot (g^{x_z j + x_h})^{t_j} \cdot (g^{\psi(j-j^*)})^{t'_j - \tau(r_j - x_d)/(j-j^*)} \\
&= (u^{r_j} d)^\tau \cdot (g^{x_z j + x_h})^{t_j} \cdot (g^{\psi(j-j^*)})^{t_j} \\
&= (u^{r_j} d)^\tau \cdot ((g^{\psi + x_z})^j g^{-\psi j^* + x_h})^{t_j} \\
&= (u^{r_j} d)^\tau \cdot (g_2^j h)^{t_j}
\end{aligned}$$

Answering Queries. Upon receiving a query from $\hat{\mathsf{R}}$, $\mathcal{B}$ verifies the accompanying WIPoK and extracts $(\sigma_i, x, \bar{c}_4, \bar{c}_5, \bar{c}_6)$ and the value v_1. Note that $\hat{\mathsf{R}}$ must issue at least one forged query where v_1/g^x is not equal to the first element of C_{σ_i} . When this occurs, if $\sigma_i \neq j^*$ then $\mathcal{B}$ aborts and outputs a random bit.

Otherwise let us consider the distribution of $\hat{\mathsf{R}}$'s query. For some $t, r' \in \mathbb{Z}_p$ the soundness of the WIPoK ensures that $(v_1/g^x, \bar{c}_6) = (g^{r'}, u^{r'})$ and $(\bar{c}_4, \bar{c}_5) = (g^t, (u^{r'} d)^a (g_2^{\sigma_i} h)^t)$. By substitution we obtain:

$$\bar{c}_5 \;=\; (g^{\psi r'} g^{-\psi x_d + y_d})^\tau (g^{(\psi + x_z) j^*} g^{-\psi j^*} g^{x_h})^t$$

$$= \quad g^{\tau\psi(r'-x_d)} g^{\tau y_d} g^{t(x_z j^* + x_h)}$$

Let us implicitly define the value $h' = (v_1/g^x)g^{-x_d} = g^{r'-x_d}$. $\mathcal{B}$ can obtain $h'^{\tau\psi}$ by computing $\bar{c}_5/(g^{\tau y_d}\bar{c}_4^{x_z j^* + x_h})$. Provided that $h' \neq 1$, $\mathcal{B}$ can now compute a solution to the 3DDH problem by comparing $e(h'^{\tau\psi}, g^{\omega}) \stackrel{?}{=} e(Z, h')$. If $h' = 1$ then $\mathcal{B}$ aborts and outputs a random bit.

Probability of abort. There are two conditions in which $\mathcal{B}$ aborts: (1) when $\hat{\mathsf{R}}$ does not issue a forgery for $\sigma_i = j^*$, and (2) when $\sigma_i = j^*$ but $(v_1/g^x)g^{-x_d} = 1$. Since j^*, x_d are outside of $\hat{\mathsf{R}}$'s view and our base assumption is that $\hat{\mathsf{R}}$ that makes at least one request on $\sigma_i \in [1, N]$, the probability that $\mathcal{B}$ does not abort is $\geq \frac{p-1}{p} \cdot \frac{1}{N}$. Thus, if no p.p.t. algorithm solves 3DDH with probability $> \epsilon$, then Adv [Game 2]- Adv [Game 1] $\leq \frac{Np\epsilon}{p-1}$. □

Lemma 5.4 *If no p.p.t adversary has advantage $> \epsilon$ at solving the 3DDH problem, then*

$$\text{Adv [Game 3]} - \text{Adv [Game 2]} \leq N\epsilon.$$

Proof. See the proof of Lemma A.3 in Ref.[8]. □

Receiver Security. For any real-world cheating sender $\hat{\mathsf{S}}$ we can construct an ideal-world sender $\hat{\mathsf{S}}'$ such that all p.p.t. distinguishers have negligible advantage at distinguishing the distribution of the real and ideal experiments. Let us now describe the operation of $\hat{\mathsf{S}}'$, which runs $\hat{\mathsf{S}}$ internally, interacting with it in the role of the Receiver.

1. To begin, $\hat{\mathsf{S}}'$ runs the $\mathsf{R_I}$ algorithm, with the following modification: when $\hat{\mathsf{S}}$ proves knowledge of a, $\hat{\mathsf{S}}'$ uses the knowledge extractor E_1 to extract a, outputting error if the extractor fails. Otherwise, it has obtained the values $(pk, C_1, \cdots, C_N)$.
2. For $i = 1$ to N, $\hat{\mathsf{S}}'$ decrypts each of $\hat{\mathsf{S}}$'s ciphertexts $C_1, \cdots, C_N$ using the value a as a decryption key, and sends the resulting $M_1^*, \cdots, M_N^*$ to the trusted party T.
3. Whenever T indicates to $\hat{\mathsf{S}}'$ that a transfer has been initiated, $\hat{\mathsf{S}}'$ runs the transfer protocol with $\hat{\mathsf{S}}$ on the fixed index 1. If the transfer succeeds, $\hat{\mathsf{S}}'$ returns the bit 1 (indicating success) to T, or 0 otherwise.
4. $\hat{\mathsf{S}}'$ uses $\hat{\mathsf{S}}$'s output as its own.

Theorem 5.5 *Let ϵ_{WI} be the maximum advantage that any p.p.t. algorithm has at distinguishing a WIPoK, and let ϵ_{Ext} be the maximum probability that the extractor E_1 fails. Then $\forall$ p.p.t. D:*

$$\Pr[D(\text{Real}_{\hat{\mathsf{S}},\mathsf{R}}(N, k, M_1, \cdots, M_N, \Sigma)) = 1] - \Pr[D(\text{Ideal}_{\hat{\mathsf{S}}',\mathsf{R}'}(N, k, M_1, \cdots, M_N, \Sigma)) = 1] \leq (k+1)\epsilon_{Ext} + k\epsilon_{WI}.$$

Proof. See the proof of Theorem 3.3 in Ref.[8]. □

6 Conclusion

In this paper, we present an improvement of one adaptive OT scheme which is based on 3DDH assumption in bilinear groups. We show that in the original scheme there are some redundancies. Using the modified simulation, we prove that the improvement keeps secure under 3DDH assumption. We believe the skills developed in the paper is helpful for optimizing other cryptographic schemes.

References

[1] Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56-73. Springer, Heidelberg (2004)

[2] Boneh, D., Sahai, A., Waters, B.: Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006, LNCS, vol. 4004, pp. 573-592. Springer, Heidelberg (2006)

[3] Camenisch, J., Neven, G., Shelat, A.: Simulatable Adaptive Oblivious Transfer. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 573-590. Springer, Heidelberg (2007)

[4] Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: CRYPTO'97, LNCS, vol. 1296, pp. 410-424. Springer, Heidelberg (1997)

[5] Chu, C.-K., Tzeng, W.-G.: Efficient oblivious transfer schemes with adaptive and non-adaptive queries. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 172-183. Springer, Heidelberg (2005)

[6] Green, M., Hohenberger, S.: Blind Identity-Based Encryption and Simulatable Oblivious Transfer. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 265-282. Springer, Heidelberg (2007)

[7] Green, M., Hohenberger, S.: Universally Composable Adaptive Oblivious Transfer. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 179-197. Springer, Heidelberg (2008)

[8] Green, M., Hohenberger, S.: Practical adaptive oblivious transfer from simple assumptions. In: Yuval, I., (Ed.) TCC 2011. LNCS, vol. 6597, pp. 347-363. Springer, Heidelberg (2011) [Cryptology ePrint Archive, Report 2010/109, http://eprint.iacr.org/]

[9] Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: STOC '87, pp. 218-229 (1987)

[10] Hohenberger, S., Waters, B.: Realizing Hash-and-Sign Signatures under Standard Assumptions. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 333-350. Springer, Heidelberg (2009)

[11] Jarecki, S., Liu, X.: Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577-594. Springer, Heidelberg (2009)

[12] Kurosawa, K., Nojima, R.: Simple Adaptive Oblivious Transfer without Random Oracle. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 334-346. Springer, Heidelberg (2009)

[13] Kurosawa, K., Nojima, R., Le Phong, T.: Efficiency-Improved Fully Simulatable Adaptive OT under the DDH Assumption. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 172-181. Springer, Heidelberg (2010)

[14] Kurosawa, K., Nojima, R., Le Phong, T.: Generic fully simulatable adaptive oblivious transfer. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 274-291. Springer, Heidelberg (2011)

[15] Lipmaa, H.: An oblivious transfer protocol with log-squared communication. In: Zhou, J., et al (eds.) ISC 2005. LNCS, vol. 3650, pp. 314-328. Springer, Heidelberg (2005)

[16] Rabin, M.: How to exchange secrets by oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University (1981)

[17] Rial, A., Kohlweiss, M., Preneel, B.: Universally composable adaptive priced oblivious transfer. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 231-247. Springer, Heidelberg (2009)

[18] Yao, Y.: How to generate and exchange secrets. In: FOCS, pp. 162-167 (1986)