Threshold-Oriented Optimistic Fair Exchange

Yang Wang[1], Man Ho Au[1], Joseph K. Liu[2],
Tsz Hon Yuen[3], and Willy Susilo[1,⋆]

[1] Centre for Computer and Information Security Research, School of Computer Science and Software Engineering, University of Wollongong, Australia
[2] Cryptography and Security Department, Institute for Infocomm Research, Singapore
[3] Department of Computer Science, University of Hong Kong, Hong Kong
yw990@uowmail.edu.au, {aau,wsusilo}@uow.edu.au, ksliu@i2r.a-star.edu.sg, thyuen@cs.hku.hk

Abstract. Fair exchange protocol aims to allow two parties to exchange digital items in a fair manner. It is well-known that fairness can only be achieved with the help of a trusted third party, usually referred to as arbitrator. A fair exchange protocol is optimistic if the arbitrator is not involved in the normal execution of the fair exchange process. That is, its presence is necessary only when one of the exchanging parties is dishonest. Traditionally, the items being exchanged are digital signatures. In this paper, we consider the items to be threshold signatures. Specifically, the signatures are created by a subset of legitimate signers instead of a single signer. We define a security model for this new notion, and provide an concrete instantiation. Our instantiation can be proven secure in the random oracle model. Our definition covers the case when the item being exchanged is a secret key of an identity-based encryption where the master secret key is split amongst a set of authorities.

1 Introduction

Optimistic fair exchange (OFE), first introduced by Asokan, Schunter and Waidner [1], is a kind of protocols aiming to guarantee fairness for two parties exchanging digital items. In OFE, a trusted third party named "arbitrator" is needed but only involved when there is a dispute between the participants. Traditionally the digital items of interest are digital signatures and the optimistic fair exchange of digital signatures constitutes an important part of any business transaction. Typically such a protocol comprises three message flows. First Alice the signer initiates the exchange by sending a partial signature to the receiver, say Bob. The partial signature serves as a commitment assuring Bob of Alice's full signature at the end of the protocol. After verifying the validity of Alice's partial signature, Bob sends its full signature to Alice in the second message flow. Later, Alice should send her full signature back to Bob and complete the exchange. In the case there is a network failure or Alice attempts to cheat by

⋆ This work is supported by ARC Future Fellowship FT0991397.

J. Lopez, X. Huang, and R. Sandhu (Eds.): NSS 2013, LNCS 7873, pp. 424–438, 2013.

refusing to send her own full signature, Bob can ask the arbitrator to make a resolution with Alice's partial signature and his own full signature. In this case the arbitrator will convert Alice's partial signature into a full one and send it back to Bob. Note that at the end of this exchange, either both Alice and Bob gain the other's full signature, or neither does. Thus the exchange is fair.

1.1 Related Work

As a useful tool in applications such as contract signing, electronic commerce and even peer-to-peer file sharing, OFE has been extensively researched since its introduction. There are several approaches in the construction of OFE, including schemes based on verifiably encrypted signatures [2,6,5,16,21,19], and sequentially two-party multisignatures [8]. It was further showed that OFE can be constructed from OR signature [7], and conventional signatures and ring signatures [12]. Some desirable properties such as setup-free [22], stand-alone [22], abuse-free [9], signer ambiguity [11], resolution ambiguity [17] and accountability [13] are proposed in literatures as well.

In [3] and [15], OFE employing multiple arbitrators are discussed to reduce the trust placed on the single arbitrator. Unfortunately, the existing techniques are either expensive or rely on synchronized clocks, which is undesirable as achieving synchronization in a peer-to-peer setting in which the arbitrators do not even know each other is hard.

Most of the previous works on OFE are done in the individual setting, in which the two involving parties are individual users and they represent themselves. An interesting scenario in OFE is that either party consists of a group of users. In such a scenario, every single user in the group can represent its party to execute transactions with another party. In [18], the authors employ a ring signature such that all the group of users' public keys are involved in the ring to ensure that each signer can sign on behalf of the party. Later, optimistic fair exchange of group signatures is considered in [10].

To the best of our knowledge, there is no previous work on OFE discussing about the scenario that only a least number of users together can represent a party. That is, for a party involving a group of n users, only at least t users of them together can sign on behalf of the party and make exchanges with other parties. We introduce the notion of *threshold-oriented optimistic fair exchange* (TOFE), which in essence is optimistic fair exchange of threshold signatures. This can be viewed as a natural way to reduce the trust placed on every single user of the group.

Besides, TOFE has other practical applications. For example, consider the case in which two parties intend to exchange a secret key of an identity-based encryption (IBE) [4]. In an identity-based setting, the key generation centre (KGC) is a high value target to adversaries as compromising the master key will break the whole system. Thus the master key is typically split amongst a set of authorities so that only when a threshold of authorities together can create a secret key for an identity [14]. Remember that the secret key of an identity can be viewed as a digital signature on the user's identity from the KGC [4]. Thus,

fair exchange of secret key of an identity-based encryption also falls within the model of OFE. In case when the master key is split amongst a set of authorities and two KGCs, perhaps each for a certain geographic location, would like to exchange a secret key of a specific identity, TOFE would be useful.

The table below summarizes the categories of exchanged digital items that have been discussed in the literatures.

Table 1. digital items that are exchanged in OFE

Schemes	Digial Items Exchanged
traditional OFE	individual signatures
Qu et al. [18]	ring signatures
Huang et al. [10]	group signatures
Our Scheme	threshold signatures / secret keys of an IBE

1.2 Contribution

In this paper, we study optimistic fair exchange in a threshold-oriented setting. Specifically, we present a formal definition for TOFE. We propose a concrete construction and demonstrate that our construction is secure in the random oracle model.

Organization. The rest of the paper is organized as follows. In Section 2, we review notations and technical preliminaries. In Section 3, the syntax of TOFE and its security definitions are presented. We present our construction and prove the security of our construction under well-known assumptions in Section 4. Finally, we conclude our paper in Section 5.

2 Preliminary

If n is a positive integer, we use $[n]$ to denote the set $\{1, \ldots, n\}$. If p is a prime, we use $\mathbb{Z}_p$ to denote the set $\{0, \ldots, p-1\}$ and $\mathbb{Z}_p^*$ to denote the set $\{a | a \in \mathbb{Z}_p \wedge \gcd(a, p) = 1\}$.

2.1 Bilinear Pairing

Let $\mathbb{G}$, $\mathbb{G}_T$ be two cyclic groups such that $|\mathbb{G}| = |\mathbb{G}_T| = p$. We say that $\hat{e}$ is a bilinear map if $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ possesses the following properties.

- the group operation in $\mathbb{G}$ and the map $\hat{e}$ are both efficiently computable.
- For all elements of $g, h \in \mathbb{G}$, $a, b \in \mathbb{Z}_p$, it holds that

$$\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$$

- There exists $g, h \in \mathbb{G}$ such that $\hat{e}(g, h)$ is not the identity element of $\mathbb{G}_T$.

2.2 Number-Theoretic Assumptions

We review the following well-known computational assumptions.

Definition 1 (DL Assumption). *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. The discrete logarithm assumption states that given a tuple $(g, Z) \in (\mathbb{G}, \mathbb{G})$, it is computationally infeasible to compute the value $z \in \mathbb{Z}_p$ such that $Z = g^z$.*

Definition 2 (CDH Assumption). *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. The computational Diffie-Hellman assumption states that given a tuple $(g, g^a, g^b) \in (\mathbb{G}, \mathbb{G}, \mathbb{G})$, it is computationally infeasible to compute the value g^{ab}.*

2.3 Secret Sharing

We review the principle of the well-known Shamir secret sharing scheme [20] here. Roughly speaking, a secret sharing scheme allows a user to divide a secret into n pieces, called shares, so that any t share holders together can recover the secret. The major idea is that it takes t points to define a polynomial, say, $f(x)$ of degree $t-1$. One could generate f in such a way that $f(0)$ is the secret to be shared. Each share is then a point $(i, f(i))$. Now with t points, one could recover the polynomial and thus the value $f(0)$. On the other hand, with only $t-1$ points, nothing about $f(0)$ would be revealed since there are exponentially many curves that pass through those $t-1$ points.

Preparation Let x be the secret to be shared. Randomly pick a polynomial f of degree $t-1$ such that $f(0) = x$. Each share is defined as $(i, f(i))$ for $i = 1$ to n.

Reconstruction One could make use of Lagrange interpolation to recover the value $f(0)$ when t points are given.

- Let $\mathcal{I}$ be a set such that $|\mathcal{I}| = t$ and that for all $i \in \mathcal{I}$, $f(i)$ is known.
- The Lagrange polynomial interpolation technique states that

$$f(x) := \sum_{i \in \mathcal{I}} f(i)\lambda_i(x),$$

  where $\lambda_i(x)$, called the Lagrange basis polynomials, is defined as

$$\lambda_i(x) := \prod_{j \in \mathcal{I} \setminus \{i\}} \frac{x-j}{i-j}.$$

  Since we are interested in $f(0)$ in the secret sharing scheme, we use λ_i to denote the value of $\lambda_i(0)$ and refer to it as the Lagrange coefficient.
- Thus, to recover the secret, one first computes the Lagrange coefficient λ_i as

$$\lambda_i := \prod_{j \in \mathcal{I} \setminus \{i\}} \frac{-j}{i-j}.$$

- Then, $f(0)$ can be recovered as

$$f(0) := \sum_{i \in \mathcal{I}} f(i)\lambda_i.$$

3 Definition of TOFE

3.1 Syntax

We adapt the definitions and security models of OFE from various literatures for our TOFE. For efficiency consideration, our definition of TOFE consists of non-interactive algorithms only. The following is the syntax of a construction of TOFE, which consists of seven algorithms. In addition, we adopt the common reference string model.

- *Common Reference String Generation* On input a security parameter 1^k, this algorithm outputs a common reference string param_{CRS} which includes the security parameter 1^k. We assume param_{CRS} is an implicit input to all algorithms described below.
- $(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{AGen}()$ This algorithm outputs the arbitrator key pairs (pk_A, sk_A).
- $(\mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i=1}^n) \leftarrow \mathsf{UGen}(n, t)$ This algorithm takes as input the required number of signers n, the threshold t and output the public key of the user pk_U, together with n secret signing keys for the signers $\mathsf{sk}_{U,i}$.
- $\mathsf{PSign} = (\mathsf{PSign}_{(s)}, \mathsf{PSign}_{(v)}, \mathsf{PSign}_{(g)})$ This is a suite of three algorithms which allows a subset of signers to create a partial signature.
  - $\hat{\sigma}_i \leftarrow \mathsf{PSign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ On input the public key of the arbitrator pk_A, a message M and a secret signing key of signer i, this algorithm outputs a partial signature share for signer i.
  - $\mathit{valid}/\mathit{invalid} \leftarrow \mathsf{PSign}_{(v)}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma}_i, i)$ On input the public key of the arbitrator pk_A and that of the user pk_U, a message M, a partial signature share $\hat{\sigma}_i$ from signer i, this algorithm checks the validity of the partial signature share created by signer i.
  - $\hat{\sigma} \leftarrow \mathsf{PSign}_{(g)}(\mathsf{pk}_A, \mathsf{pk}_U, M, \{\hat{\sigma}_i\}_{i\in\mathcal{I}}, \mathcal{I})$ On input the public key of the arbitrator pk_A and that of the user pk_U, a message M, t partial signature shares $\{\hat{\sigma}_i\}$ for $i \in \mathcal{I}$ such that $\mathcal{I} \subset [n]$ and $|\mathcal{I}| = t$, this algorithm outputs a partial signature.
- $\mathit{valid}/\mathit{invalid} \leftarrow \mathsf{PVer}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma})$ This algorithm checks the validity of a partial signature $\hat{\sigma}$ on message M based on the public key of the arbitrator pk_A, the public key of the user pk_U.
- $\mathsf{Sign} = (\mathsf{Sign}_{(s)}, \mathsf{Sign}_{(v)}, \mathsf{Sign}_{(g)})$ Similar to the partial signature generation process, the signing algorithm is also a set of three algorithms which allows a subset of signers to create a signature.
  - $\sigma_i \leftarrow \mathsf{Sign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ On input public key of the arbitrator pk_A, message M and secret signer key of signer i, this algorithm outputs a signature share for signer i.
  - $\mathit{valid}/\mathit{invalid} \leftarrow \mathsf{Sign}_{(v)}(\mathsf{pk}_A, \mathsf{pk}_U, M, \sigma_i, i)$ This algorithm checks the validity of the signature share σ_i created by signer i based on the public key of the arbitrator pk_A, the public key of the user pk_U and message M.

  - $\sigma \leftarrow \mathsf{Sign}_{(g)}(\mathsf{pk}_A, \mathsf{pk}_U, M, \{\sigma_i\}_{i\in\mathcal{I}}, \mathcal{I})$ On input the public key of the arbitrator pk_A and that of the user pk_U, a message M, t signature shares $\{\sigma_i\}$ for $i \in \mathcal{I}$ such that $\mathcal{I} \subset [n]$ and $|\mathcal{I}| = t$, this algorithm outputs a signature.
- $\boldsymbol{valid}/\boldsymbol{invalid} \leftarrow \mathsf{Ver}(\mathsf{pk}_A, \mathsf{pk}_U, M, \sigma)$ This algorithm checks the validity of a signature σ on message M based on the public key of the arbitrator pk_A and that of the user pk_U.
- $\sigma \leftarrow \mathsf{Res}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma}, \mathsf{sk}_A)$ Given a valid partial signature $\hat{\sigma}$, a message M, public key of the user pk_U, key pair of the arbitrator $(\mathsf{pk}_A, \mathsf{sk}_A)$, this algorithm allows the arbitrator to output a signature on message M. Note that $\perp$ is returned if $\texttt{invalid} \leftarrow \mathsf{PVer}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma})$.

Correctness. A construction of TOFE is correct if the following conditions hold:

1. Any partial signature created by any t honest signers using PSign will be valid under PVer.
2. Any signature created by any t honest signers using Sign will be valid under Ver.
3. Any signature created by the arbitrator using Res based on a valid partial signature will be valid under Ver.

Furthermore, it is required that any signature created by the arbitrator using Res based on a valid partial signature will be indistinguishable from the signature created by any t honest signers using Sign.

3.2 A Typical Usage of the TOFE Algorithms

Note that in OFE with three message flows between the initiator Alice and the receiver Bob, the item to be sent by Bob is not restricted to any format. It could be a digital item such as electronic money. For simplicity we assume the item to be sent by Bob is a digital signature. Nonetheless, it could be a ring signature, a group signature or a threshold signature. Below we show how Alice and Bob can conduct an exchange based on our definition of TOFE. Note that the party Alice in TOFE consists of a group of n signers, and an exchange is possible only when at least t-out-of-n signers agree to participate.

Our definition of TOFE does not require the set of t signers to communicate with each other. Below is a typical usage of our definition of TOFE algorithms.

1. *Partial Signature Shares Collection* Bob approaches each signer independently and the signers agree on the items to be exchanged. The signer, say signer i, invokes $\mathsf{PSign}_{(s)}$ and sends the share of the partial signature $\hat{\sigma}_i$ to Bob. Bob uses $\mathsf{PSign}_{(v)}$ to verify the share.
2. *Partial Signature Generation* Upon collecting t partial signature shares, Bob invokes $\mathsf{PSign}_{(g)}$ to generate a partial signature $\hat{\sigma}$. He invokes PVer to ensure its validity.
3. *Obligation Fulfillment* If the partial signature Bob obtained is valid, he fulfills his obligations. In this example, Bob sends his digital signature to all the signers involved.

4. *Signature Shares Collection* Each signer validates that Bob has fulfilled his obligations. In this example, each signer checks that the digital signature sent by Bob is valid. If yes, each signer, say signer i, invokes $\mathsf{Sign}_{(s)}$ and sends the share of the signature σ_i to Bob, who checks its validity with $\mathsf{Sign}_{(v)}$.
5. *Signature Generation* Upon collecting t signature shares, Bob invokes $\mathsf{Sign}_{(g)}$ to generate a signature σ. He invokes Ver to ensure its validity. If yes, the exchange process is completed.
6. *Resolution* Suppose some signers refuse to send their signature shares, or that the signature created in signature generation is invalid, Bob can approach the arbitrator for assistance. Specifically, he approaches the arbitrator and proves that he has fulfilled his obligation. After that, Bob submits the valid partial signature $\hat{\sigma}$ to the arbitrator. The arbitrator sends back the signature σ by invoking Res and this completes the exchange.
7. *Remarks* In this example, Bob can send his digital signature to the arbitrator as a proof of obligation fulfillment. Even if Bob is lying, the arbitrator can still give this digital signature to the signers should they also complain and thus the exchange could be completed regardless of what happens afterwards.

3.3 Security Model

Traditionally, any construction of optimistic fair exchange should be secure in three aspects, namely, security against signers, security against verifiers and security against the arbitrator respectively. As suggested by the respective names, they intend to cover the scenarios when the named party is dishonest. We modify the traditional model in the threshold setting. Specifically, the verifier can collude with $t-1$ malicious signers in our consideration of security against verifiers.

Security against Signers. This property guarantees that even when all the signers collude together, they cannot create a partial signature that passes the partial signature verification algorithm PVer yet it cannot be resolved into a full signature by the arbitrator. This property intends to protect honest verifiers. Specifically, we use the following three-phase game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ to define this property.

Initialization $\mathcal{A}$ specifies the number of signers n and the threshold t. $\mathcal{C}$ creates the common reference string param_{CRS} and invokes

$$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{AGen}(),$$

$$(\mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i=1}^{n}) \leftarrow \mathsf{UGen}(n, t).$$

$\mathcal{C}$ gives $(\mathsf{param}_{CRS}, \mathsf{pk}_A, \mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i=1}^{n})$ to $\mathcal{A}$.

Query $\mathcal{A}$ can adaptively issue the following query to $\mathcal{C}$.

- Res Query. $\mathcal{A}$ gives $(\hat{\sigma}, M)$ to $\mathcal{C}$, who invokes

$$\sigma \leftarrow \mathsf{Res}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma}, \mathsf{sk}_A)$$

and returns σ to $\mathcal{A}$.

End-Game $\mathcal{A}$ submits $(M^*, \hat{\sigma}^*)$ and wins the game if

$$\mathtt{valid} \leftarrow \mathsf{PVer}(\mathsf{pk}_A, \mathsf{pk}_U, M^*, \hat{\sigma}^*)$$

$$\mathtt{invalid} \leftarrow \mathsf{Ver}\big(\mathsf{pk}_A, \mathsf{pk}_U, M^*, \mathsf{Res}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma}^*, \mathsf{sk}_A)\big)$$

Security against Verifiers. This property guarantees that even when the verifier colludes with $t-1$ signers, they cannot create a valid full signature. This property intends to protect honest signers. Our model is static in the sense that the subset of signers to be controlled by the attacker is fixed during the initialization phase. Specifically, we use the following three-phase game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ to define this property.

Initialization $\mathcal{A}$ specifies the number of signers n and the threshold t, together with an index set $\mathcal{I}' \subset [n]$ such that $|\mathcal{I}'| = t-1$. $\mathcal{C}$ creates the common reference string param_{CRS} and invokes

$$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{AGen}(),$$

$$(\mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i=1}^n) \leftarrow \mathsf{UGen}(n, t).$$

$\mathcal{C}$ gives $(\mathsf{param}_{CRS}, \mathsf{pk}_A, \mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i\in\mathcal{I}'})$ to $\mathcal{A}$.

Query $\mathcal{A}$ can adaptively issue the following query to $\mathcal{C}$.

- $\mathsf{PSign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{C}$, who invokes $\hat{\sigma}_i \leftarrow \mathsf{PSign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ and returns $\hat{\sigma}_i$ to $\mathcal{A}$.
- $\mathsf{Sign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{C}$, who invokes $\sigma_i \leftarrow \mathsf{Sign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ and returns σ_i to $\mathcal{A}$.
- Res Query. $\mathcal{A}$ gives $(\hat{\sigma}, M)$ to $\mathcal{C}$, who invokes $\sigma \leftarrow \mathsf{Res}(\mathsf{pk}_A, \mathsf{pk}_U, M, \hat{\sigma}, \mathsf{sk}_A)$ and returns σ to $\mathcal{A}$.

End-Game $\mathcal{A}$ submits $(M^*, \hat{\sigma}^*)$ and wins the game if

$$\mathtt{valid} \leftarrow \mathsf{Ver}(\mathsf{pk}_A, \mathsf{pk}_U, M^*, \hat{\sigma}^*)$$

and that $(M^*, \cdot)$ did not appear in any $\mathsf{Sign}_{(s)}$ query. Furthermore, if there exists a $\mathsf{PSign}_{(s)}$ query with input $(M^*, \cdot)$, $(\cdot, M^*)$ should not appear as input in any Res query.

Security against the Arbitrator. This property guarantees that the arbitrator cannot create a signature on behalf of the user unless it is given a valid partial signature. In TOFE, we allow the arbitrator to collude with $t-1$ signers. As in the case of security against verifiers, our model is static in the sense that the subset of signers to be controlled by the attacker is fixed during the initialization phase. Specifically, we use the following three-phase game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ to define this property.

Initialization $\mathcal{A}$ specifies the number of signers n and the threshold t, together with an index set $\mathcal{I}' \subset [n]$ such that $|\mathcal{I}'| = t - 1$. $\mathcal{C}$ creates the common reference string param_{CRS} and invokes

$$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{AGen}(),$$

$$(\mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i=1}^{n}) \leftarrow \mathsf{UGen}(n, t).$$

$\mathcal{C}$ gives $(\mathsf{param}_{CRS}, \mathsf{pk}_A, \mathsf{pk}_U, \{\mathsf{sk}_{U,i}\}_{i\in\mathcal{I}'}, \mathsf{sk}_A)$ to $\mathcal{A}$.

Query $\mathcal{A}$ can adaptively issue the following query to $\mathcal{C}$.

- $\mathsf{PSign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{C}$, who invokes $\hat{\sigma}_i \leftarrow \mathsf{PSign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ and returns $\hat{\sigma}_i$ to $\mathcal{A}$.
- $\mathsf{Sign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{C}$, who invokes $\sigma_i \leftarrow \mathsf{Sign}_{(s)}(\mathsf{pk}_A, M, \mathsf{sk}_{U,i})$ and returns σ_i to $\mathcal{A}$.

End-Game $\mathcal{A}$ submits $(M^*, \hat{\sigma}^*)$ and wins the game if

$$\mathtt{valid} \leftarrow \mathsf{Ver}(\mathsf{pk}_A, \mathsf{pk}_U, M^*, \hat{\sigma}^*)$$

and that $(M^*, \cdot)$ did not appear in any $\mathsf{Sign}_{(s)}$ query nor $\mathsf{PSign}_{(s)}$ query.

4 Construction

Our TOFE is motivated by the ordinary OFE by [5]. Indeed, when $t = n = 1$, our construction degenerates to their scheme.

Common Reference String Our construction works in the common reference string model. For a security parameter 1^k, let $\mathbb{G}$, $\mathbb{G}_T$ be cyclic groups of prime order p with g as a generator of $\mathbb{G}$, where p is a k-bit prime. Further, let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. The common reference string is defined to be

$$\mathsf{param}_{CRS} := (1^k, \mathbb{G}, \mathbb{G}_T, p, g, \hat{e}).$$

AGen On input param_{CRS}, the arbitrator picks at random $y \in_R \mathbb{Z}_p$ and computes $Y = g^y$. The public key and secret key of the arbitrator is defined as

$$(\mathsf{pk}_A, \mathsf{sk}_A) := \big(Y, y\big).$$

UGen On input param_{CRS}, the required number of signers n and the threshold t, the user picks at random a polynomial of degree $t-1$ in $\mathbb{Z}_p$, say f. Assume the signers are indexed by i, for $i = 1$ to n, with $n \geq t \geq 1$. The user further picks at random a hash function $H : \{0,1\}^* \to \mathbb{G}$. Note that H is to be modelled as a random oracle.

For $i = 1$ to n, the secret signing key of signer i is defined as $f(i)$.

The user computes the public key as

$$\mathsf{pk}_U := (H, X, X_1, \ldots, X_n) := (H, g^{f(0)}, g^{f(1)}, \ldots, g^{f(n)}).$$

The value $f(0)$, which is the actual master secret, should be deleted. This ensures only a set of t signers together could create a threshold signature.

PSign. The partial signature generation process consists of three sub-algorithms.

- *Generation of a Partial Signature Share* On input param_{CRS}, pk_A, a message M and the signing key of signer i $f(i)$, signer i randomly picks $r_i \in_R \mathbb{Z}_p$ and outputs the partial signature share as
$$\hat{\sigma}_i := (\alpha_i, \beta_i) := \big(H(M)^{f(i)} Y^{r_i}, g^{r_i}\big).$$
- *Verification of a Partial Signature Share* The partial signature share $\hat{\sigma}_i$ can be verified by evaluating the following relation:
$$\hat{e}(\alpha_i, g) \stackrel{?}{=} \hat{e}(H(M), X_i)\hat{e}(Y, \beta_i).$$
- *Generation of a Partial Signature* When t partial signature shares, say, $\hat{\sigma}_i$ for $i \in \mathcal{I} \subset [n]$ such that $|\mathcal{I}| = t$ on the same message, say M, have been collected, anyone can output the partial signature on message M as:
$$\hat{\sigma} := (\alpha, \beta) := (\prod_{i\in\mathcal{I}} \alpha_i^{\lambda_i}, \prod_{i\in\mathcal{I}} \beta_i^{\lambda_i}).$$
where λ_i is defined as
$$\lambda_i := \prod_{j\in\mathcal{I}\setminus\{i\}} \frac{-j}{i-j}.$$
As discussed, $f(0) = \sum_{i\in\mathcal{I}} f(i)\lambda_i$.

PVer. On input param_{CRS}, pk_A, pk_U, a message M and a partial signature $\hat{\sigma}$, the algorithm outputs `valid` if and only if the following equality holds:
$$\hat{e}(\alpha, g) = \hat{e}\big(H(M), X\big)\hat{e}(Y, \beta).$$

Sign. The full signature generation process consists of three sub-algorithms as well.

- *Generation of a Signature Share* On input param_{CRS}, pk_A, a message M and the signing key of signer i $f(i)$, signer i outputs the signature share as
$$\sigma_i := H(M)^{f(i)}.$$
- *Verification of a Signature Share* The signature share σ_i can be verified by evaluating the following relation:
$$\hat{e}(\sigma_i, g) \stackrel{?}{=} \hat{e}(H(M), X_i).$$
- *Generation of a Signature* When t signature shares, say, σ_i for $i \in \mathcal{I} \subset [n]$ such that $|\mathcal{I}| = t$ on the same message, say M, have been collected, anyone can output the signature on message M as:
$$\sigma := \prod_{i\in\mathcal{I}} \sigma_i^{\lambda_i}$$
where λ_i is defined as
$$\lambda_i := \prod_{j\in\mathcal{I}\setminus\{i\}} \frac{-j}{i-j}.$$

Ver. On input param_{CRS}, pk_A, pk_U, a message M and a signature σ, the algorithm outputs `valid` if and only if the following equality holds:

$$\hat{e}(\sigma, g) = \hat{e}(H(M), X).$$

Res. On input param_{CRS}, pk_A, pk_U, a message M, a partial signature $\hat{\sigma}$ and the secret key of the arbitrator y, the full signature can be computed as follows.

- Check that $\hat{\sigma}$ is a valid partial signature by evaluating the relation

$$\hat{e}(\alpha, g) \stackrel{?}{=} \hat{e}(H(M), X)\hat{e}(Y, \beta).$$

- Output σ as

$$\sigma := \alpha/\beta^y.$$

Regarding the security of our construction of TOFE, we have the following theorem.

Theorem 1. *Our construction of TOFE is secure against signers, verifiers and the arbitrator under the CDH assumption in the random oracle model.*

Proof. Security against signers. Given a valid partial signature $\hat{\sigma}^* := (\alpha^*, \beta^*)$ on message M^*, such that

$$\hat{e}(\alpha^*, g) = \hat{e}(H(M^*), X)\hat{e}(Y, \beta^*),$$

the resolved signature σ is defined as $\alpha^*/(\beta^*)^y$ where $Y = g^y$.

Note that

$$\hat{e}(\sigma, g) = \frac{\hat{e}(\alpha^*, g)}{\hat{e}((\beta^*)^y, g)} = \frac{\hat{e}(H(M^*), X)\hat{e}(Y, \beta^*)}{\hat{e}((\beta^*), g^y)} = \hat{e}(H(M^*), X),$$

any valid partial signature will always be resolved to a valid full signature.

Security against verifiers. Suppose the final output of $\mathcal{A}$ is (M^*, σ^*). If $\mathcal{A}$ has not made a $\mathsf{PSign}_{(s)}$ query with input $(M^*, \cdot)$, the analysis of this type of attack is covered in the security against the arbitrator to be discussed later. Thus without loss of generality, we safely assume that $\mathcal{A}$ has made a $\mathsf{PSign}_{(s)}$ query with input $(M^*, \cdot)$. In this setting, we show how to construct a simulator $\mathcal{S}$ that is given $A = g^a$, $B = g^b$ and tries to solve the CDH problem by outputting g^{ab}.

Initialization $\mathcal{A}$ specifies the number of signers n and the threshold t, together with an index set $\mathcal{I}' \subset [n]$ such that $|\mathcal{I}'| = t - 1$. $\mathcal{S}$ sets the common reference string param_{CRS}, $\mathsf{pk}_A = B^y$ for some randomly picked $y \in_R \mathbb{Z}_p$. For each $i \in \mathcal{I}'$, $\mathcal{S}$ picks $s_i \in_R \mathbb{Z}_p$ and computes $X_i = g^{s_i}$. $\mathcal{S}$ sets $X = g^a$. Consider a degree $t-1$ polynomial $f(x)$ such that $f(0) = a$ and $f(i) = s_i$ for $i \in \mathcal{I}'$. Note that the set of points $(0, a) \cup \{(i, s_i)\}_{i \in \mathcal{I}'}$ uniquely determines this polynomial yet the coefficients are unknown to $\mathcal{S}$. However, $\mathcal{S}$ can still compute $X_i = g^{f(i)}$ for $i \in [n] \setminus \mathcal{J}$ where $\mathcal{J} := 0 \cup \mathcal{I}'$ using the Lagrange

polynomial interpolation technique discussed in Section 2. Specifically, for $i \in [n] \setminus \mathcal{J}$,

$$g^{f(i)} = g^{\sum_{j \in \mathcal{J}} f(j)\lambda_j(i)} = \prod_{j \in \mathcal{J}} \left(g^{f(j)}\right)^{\lambda_j(i)}.$$

Note that both $\lambda_j(i)$ and $g^{f(j)}$ for all $j \in \mathcal{J}$ are computable by $\mathcal{S}$ and thus $\mathcal{S}$ can compute $X_i = g^{f(i)}$ for all $i = 1$ to n. $\mathcal{S}$ also specifies the random oracle H. pk_U is set to be $(H, X, X_1, \ldots, X_n)$. $\mathcal{S}$ gives $(\mathsf{param}_{CRS}, \mathsf{pk}_A, \mathsf{pk}_U, \{s_i\}_{i \in \mathcal{I}'})$ to $\mathcal{A}$.

Query $\mathcal{A}$ can adaptively issue the following query to $\mathcal{S}$.

- Random Oracle H Query. Suppose $\mathcal{A}$ makes q queries of this type. $\mathcal{S}$ picks an index $z \in [q]$ at random. For the h-th query, $\mathcal{A}$ submits a value M_h and is expecting the value of $H(M_h)$. If $h \neq z$, $\mathcal{S}$ replies with g^{d_h} for a random $d_h \in_R \mathbb{Z}_p$. For the z-th query, $\mathcal{S}$ replies with g^b.
- $\mathsf{PSign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{S}$. Then, $\mathcal{S}$ locates the random oracle H query for M. If there exists h such that $M = M_h$ and that $h \neq z$, $\mathcal{S}$ picks $r \in_R \mathbb{Z}_p$ at random and responses with $(\alpha, \beta) = (Y^r X_i^{d_h}, g^r)$. If M has not been queried, $\mathcal{S}$ makes such a random oracle query on input M. If $M = M_z$, $\mathcal{S}$ responses as follows.
  - Note that each X_i for $i \in [n] \setminus \mathcal{I}'$ is of the form $g^{u_i a + v_i}$ for some constant $u_i \neq 0$, v_i known by $\mathcal{S}$.
  - $\mathcal{S}$ computes r such that $u_i = -yr$.
  - $\mathcal{S}$ randomly picks $t_i \in_R \mathbb{Z}_p$, computes $\beta_i = (g^a)^r g^{t_i}$ and $\alpha_i = (g^b)^{v_i + y t_i}$ and returns (α_i, β_i) to $\mathcal{A}$.
- $\mathsf{Sign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{S}$. If $M = M_z$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ can locate h such that $H(M) = g^{d_h}$. Next, $\mathcal{S}$ computes $\sigma_i = X_i^{d_h}$ and returns σ_i to $\mathcal{A}$.
- Res Query. $\mathcal{A}$ gives $(\hat{\sigma}, M)$ to $\mathcal{S}$. $\mathcal{S}$ first checks the validity of $\hat{\sigma}$ and proceeds if it is valid. Otherwise it returns $\perp$. Then, $\mathcal{S}$ locates the random oracle H query for M. If $M = M_z$, $\mathcal{S}$ aborts. Otherwise, there exists h such that $H(M) = g^{d_h}$. Next, $\mathcal{S}$ computes $\sigma = X^{d_h}$ and returns σ to $\mathcal{A}$.

End-Game $\mathcal{A}$ submits $(M^*, \hat{\sigma}^*)$. If $M^* \neq M_z$, $\mathcal{S}$ aborts. In the random oracle model, M must have been submitted as an input in the random oracle H-query. Thus, with probability $1/q$, $\mathcal{S}$ does not abort. $\mathcal{S}$ outputs σ as the solution to the CDH problem. Note that in order to win,

$$\hat{e}(\sigma, g) = \hat{e}\big(H(M^*), X\big).$$

It implies that $\sigma = g^{ab}$.

Security against the arbitrator. We show any adversary $\mathcal{A}$ that breaks the security against the arbitrator can be converted into a simulator $\mathcal{S}$ that solves the CDH problem. $\mathcal{S}$ is given $A = g^a$, $B = g^b$ and its goal is to output g^{ab}.

Initialization $\mathcal{A}$ specifies the number of signers n and the threshold t, together with an index set $\mathcal{I}' \subset [n]$ such that $|\mathcal{I}'| = t - 1$. $\mathcal{S}$ sets the common reference string param_{CRS}, $\mathsf{pk}_A = g^y$ for some randomly picked $y \in_R \mathbb{Z}_p$.

For each $i \in \mathcal{I}'$, $\mathcal{S}$ picks $s_i \in_R \mathbb{Z}_p$ and computes $X_i = g^{s_i}$. $\mathcal{S}$ sets $X = g^a$. Consider a degree $t-1$ polynomial $f(x)$ such that $f(0) = a$ and $f(i) = s_i$ for $i \in \mathcal{I}'$. Note that the set of points $(0, a) \cup \{(i, s_i)\}_{i \in \mathcal{I}'}$ uniquely determines this polynomial yet the coefficients are unknown to $\mathcal{S}$. However, $\mathcal{S}$ can still compute $X_i = g^{f(i)}$ for $i \in [n] \setminus \mathcal{J}$ where $\mathcal{J} := 0 \cup \mathcal{I}'$ as

$$g^{f(i)} = g^{\sum_{j \in \mathcal{J}} f(j)\lambda_j(i)} = \prod_{j \in \mathcal{J}} \left(g^{f(j)}\right)^{\lambda_j(i)}.$$

$\mathcal{S}$ also specifies the random oracle H. pk_U is set to be $(H, X, X_1, \ldots, X_n)$. $\mathcal{S}$ gives $(\mathsf{param}_{CRS}, \mathsf{pk}_A, \mathsf{pk}_U, \{s_i\}_{i \in \mathcal{I}'}, y)$ to $\mathcal{A}$.

Query $\mathcal{A}$ can adaptively issue the following query to $\mathcal{S}$.

- Random Oracle H Query. Suppose $\mathcal{A}$ makes q queries of this type. $\mathcal{S}$ picks an index $z \in [q]$ at random. For the h-th query, $\mathcal{A}$ submits a value M_h and is expecting the value of $H(M_h)$. If $h \neq z$, $\mathcal{S}$ replies with g^{d_h} for a random $d_h \in_R \mathbb{Z}_p$. For the z-th query, $\mathcal{S}$ replies with g^b.
- $\mathsf{PSign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{S}$. Then, $\mathcal{S}$ locates the random oracle H query for M. If there exists h such that $M = M_h$ and that $h \neq z$, $\mathcal{S}$ picks $r \in_R \mathbb{Z}_p$ at random and responses with $(\alpha, \beta) = (Y^r X_i^{d_h}, g^r)$. If M has not been queried, $\mathcal{S}$ makes such a random oracle query on input M. If $M = M_z$, $\mathcal{S}$ aborts.
- $\mathsf{Sign}_{(s)}$ Query. $\mathcal{A}$ gives (M, i) to $\mathcal{S}$. If $M = M_z$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ can locate h such that $H(M) = g^{d_h}$. Next, $\mathcal{S}$ computes $\sigma_i = X_i^{d_h}$ and returns σ_i to $\mathcal{A}$.

End-Game $\mathcal{A}$ submits $(M^*, \hat{\sigma}^*)$. If $M^* \neq M_z$, $\mathcal{S}$ aborts. In the random oracle model, M must have been submitted as an input in the random oracle H-query. Thus, with probability $1/q$, $\mathcal{S}$ does not abort. $\mathcal{S}$ outputs σ as the solution to the CDH problem. Note that in order to win,

$$\hat{e}(\sigma, g) = \hat{e}\big(H(M^*), X\big).$$

It implies that $\sigma = g^{ab}$.

This completes the proof of the theorem. □

5 Conclusion

We present the first threshold-oriented fair exchange protocol which allows a subset of signers to exchange a digital item with a counter party. Indeed, in our specific construction, the item being exchanged is a threshold signature. We define formal security model for TOFE, present an efficient construction and show that it is secure in the random oracle model under well-known assumptions. We leave construction of TOFE in the standard model as an open problem.

References

1. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: ACM Conference on Computer and Communications Security, pp. 7–17 (1997)
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
3. Avoine, G., Vaudenay, S.: Optimistic fair exchange based on publicly verifiable secret sharing. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 74–85. Springer, Heidelberg (2004)
4. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
6. Camenisch, J., Damgård, I.: Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 331–345. Springer, Heidelberg (2000)
7. Dodis, Y., Lee, P.J., Yum, D.H.: Optimistic fair exchange in a multi-user setting. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 118–133. Springer, Heidelberg (2007)
8. Dodis, Y., Reyzin, L.: Breaking and repairing optimistic fair exchange from podc 2003. In: DRM 2003, pp. 47–54 (2003)
9. Garay, J.A., Jakobsson, M., MacKenzie, P.D.: Abuse-free optimistic contract signing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
10. Huang, Q., Wong, D.S., Susilo, W.: Group-oriented fair exchange of signatures. Inf. Sci. 181(16), 3267–3283 (2011)
11. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 74–89. Springer, Heidelberg (2008)
12. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 106–120. Springer, Heidelberg (2008)
13. Huang, X., Mu, Y., Susilo, W., Wu, W., Zhou, J., Deng, R.H.: Preserving transparency and accountability in optimistic fair exchange of digital signatures. IEEE Transactions on Information Forensics and Security 6(2), 498–512 (2011)
14. Kate, A., Goldberg, I.: Distributed private-key generators for identity-based cryptography. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 436–453. Springer, Heidelberg (2010)
15. Küpçü, A., Lysyanskaya, A.: Optimistic fair exchange with multiple arbiters. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 488–507. Springer, Heidelberg (2010)
16. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential aggregate signatures and multisignatures without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)

17. Markowitch, O., Kremer, S.: An optimistic non-repudiation protocol with transparent trusted third party. In: Davida, G.I., Frankel, Y. (eds.) ISC 2001. LNCS, vol. 2200, pp. 363–378. Springer, Heidelberg (2001)
18. Qu, L., Wang, G., Mu, Y.: Optimistic fair exchange of ring signatures. In: Rajarajan, M., Piper, F., Wang, H., Kesidis, G. (eds.) SecureComm 2011. LNICST, vol. 96, pp. 227–242. Springer, Heidelberg (2012)
19. Rückert, M., Schröder, D.: Security of verifiably encrypted signatures and a construction without random oracles. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 17–34. Springer, Heidelberg (2009)
20. Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (1979)
21. Zhang, J., Mao, J.: A novel verifiably encrypted signature scheme without random oracle. In: Dawson, E., Wong, D.S. (eds.) ISPEC 2007. LNCS, vol. 4464, pp. 65–78. Springer, Heidelberg (2007)
22. Zhu, H., Bao, F.: Stand-alone and setup-free verifiably committed signatures. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 159–173. Springer, Heidelberg (2006)