The ISC Int'l Journal of Information Security
ISeCure

January 2015, Volume 7, Number 1 (pp. 3–14)
http://www.isecure-journal.org

A Traceable Optimistic Fair Exchange Protocol in the Standard Model[☆]

Ramin Ganjavi [1,*], Maryam Rajabzadeh Asaar [1], and Mahmoud Salmasizadeh [2]
[1] *Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran*
[2] *Electronics Research Institute, Sharif University of Technology, Tehran, Iran*

ARTICLE INFO.

Article history:
Received: 14 November 2014
Revised: 23 March 2015
Accepted: 28 April 2015
Published Online: 30 April 2015

Keywords:
Optimistic Fair Exchange, Accountability, Traceable Ring Signature, Standard Model.

ABSTRACT

An Optimistic Fair Exchange (OFE) protocol is a good way for two parties to exchange their digital items in a fair way such that at the end of the protocol execution, both of them receive their items or none of them receive anything. In an OFE protocol there is a semi-trusted third party, named arbitrator, which involves in the protocol if it is necessary. But there is a security problem when arbitrator acts dishonestly and colludes with the verifier, that is, the arbitrator can complete the transaction without getting signer's agreement. Huang *et al.* in 2011 addressed this issue by formalizing the accountability property. However, Huang *et al.*'s scheme is secure in the random oracle model which is not available in the real world. We present the first generic accountable OFE protocol that is secure in the standard model by using traceable ring signatures (TRSs) as our primitive. We prove the security of our protocol under the chosen-key model and multi-user setting.

1 Introduction

An OFE protocol consists of three parties: a signer, a verifier, and an arbitrator. First, Alice, the signer, sends Bob, the verifier, her partial signature on a specific item, to ensure Bob that she wants to do the transaction. These items are electronic products or services such as software, music, video, utility bills, electronic magazines, and journals. Then, Bob sends Alice the item agreed to be exchanged. After that, Alice sends Bob her full signature and terminates the protocol. If Alice refuses to send Bob her full signature after receiving the item from Bob, Bob can ask the arbitrator to convert the partial signature to the full signature by which, the verifier can take the item's price from the signer's bank account. Because of this scenario, at the end of the protocol, both parties receive their items from each other or none of them obtains anything. The term optimistic means that the arbitrator is not always online and interferes just when a dispute occurs.

There are various paradigms to construct OFE protocols. A conventional way is verifiable encrypted signatures [2] in which the full signature is an ordinary signature. The partial signature is an ordinary signature encrypted under the arbitrator's public key along with a zero-knowledge proof which implies that the encrypted signature consists of the ordinary signature. In this type of OFE protocol, if Alice refuses to send Bob the full signature, Bob can refer to the arbitrator and wants him to decrypt Alice's partial signature.

[☆] An earlier version of this paper presented in 11th International ISC Conference on Information Security and Cryptology, Tehran, Iran [1]
[*] Corresponding author.

Email addresses: ganjavi@ee.sharif.edu (R. Ganjavi), asaar@ee.sharif.edu (M. Rajabzadeh Asaar), salmasi@sharif.edu (M. Salmasizadeh).

ISSN: 2008-2045 © 2015 ISC. All rights reserved.

Another way to construct OFE protocol is based on sequential two-party multisignatures [3]. In such a signature, two signers are able to sign a message using their private keys. Park *et al.* proposed an OFE protocol based on multisignature [4]. In park *et al.*'s scheme, the signer splits her private key into two partial parts and sends the second part to the arbitrator. Then, she signs a message with respect to the first part as a partial signature. The full signature is another signature on the partial signature, this time with respect to the second part of the private key. By using the second part of the private key, the arbitrator is always capable of transforming the partial signature to the full signature in a situation where Bob does not receive Alice's full signature.

Another paradigm for constructing OFE protocol is proposed by Huang *et al.* [5]. The partial signature in Huang *et al.*'s scheme is an ordinary signature and the full signature is a partial signature along with a ring signature in a ring which consists of the signer and the arbitrator.

Almost all OFE protocols have the resolution ambiguity (also known as transparency) property that indicates the full signatures generated by the signer should be indistinguishable from the ones created by the arbitrator. But this property can compromise the fairness of the OFE protocols if the arbitrator acts dishonest and colludes with the verifiers. As an example consider Huang *et al.*'s protocol [5] in the following scenario.

- The verifier receives the partial signature from the signer and stops the protocol without sending anything to the signer. Then, he refers to the arbitrator and asks him to convert the partial signature to the full signature. Next, the arbitrator generates a ring signature and creates the full signature. Due to the anonymity of ring signatures, no one can trace the full signature creator. As a result, the verifier achieves the full signature of the signer and can refer to the bank and draw some money out of the signer's account without sending back anything to the signer.

To solve the above mentioned problem, Huang *et al.* formalized the notion of accountable OFE to identify the dishonest party [6]. They proposed the first generic design of OFE protocol with accountability, where the identity of the full signature generator (either the signer or the arbitrator) remains unknown until certain proofs are issued.

But Huang *et al.*'s protocol is secure in the random oracle model which does not essentially remain secure in the real world, since there is not any real hash function that can implement a true random oracle. Therefore, we present the first generic accountable OFE protocol which is secure in the standard model. Our protocol is similar to Huang *et al.*'s protocol [5] with the only difference that we use traceable ring signature (TRS) as our ring signature. In a normal situation, our OFE protocol is similar to the ordinary OFE protocols, but in a situation where a dispute occurs about who is the real producer of the full signature, both the arbitrator and the signer can present their evidences, by which and according to the tracing algorithm, we can decide who the real signer is.

1.1 Related Work

Optimistic fair exchange (OFE) was first introduced by Asokan *et al.* in 1997, to solve the problem of fairly exchanging digital items [7]. Later, Asokan *et al.* formalized the notion of OFE of digital signatures [8] and proposed a concrete protocol which has the timely-termination and accountability properties. Timely-termination means that if one party does not respond for a long time, the other can ask the arbitrator to stop the protocol. Asokan *et al.*'s protocols are interactive protocol, while in 2003, Dodis and Reyzin proposed the notion of verifiably committed signatures by which the security model of non-interactive OFE protocol is formally defined [9]. Their security model consists of three aspects: security against signers, security against verifiers and security against the arbitrator. This model is setup-driven (initial key registration is required between users and the arbitrator) and single-user setting that consists of one signer, one verifier along with one arbitrator. Next in 2008, Dodis *et al.* presented a setup-free and multi-user setting scheme, that is, there are many signers and verifiers along with one arbitrator [10]. In this model, a malicious party can collude with other parties and attack the OFE protocol. Dodis *et al.* showed by a counterexample that an OFE protocol which is secure in the single-user setting is not necessarily secure in the multi-user setting [10]. Huang *et al.* defined a new property called strong resolution ambiguity which means that if one can convert a partial signature to a full signature using signer's private key or arbitrator's private key, and outputs such a pair , it is impossible to tell which key is used in the conversion [11]. They proved that for an OFE protocol with strong resolution ambiguity, the security in the single-user setting is preserved in the multi-user setting [11]. Huang *et al.* [5] considered the security model of OFE protocol not only in the multi-user setting, but also in the chosen-key model while all previous OFE protocols were in the certified-key model. The chosen-key model is in contrast to the certified-key model, in which before using a public key, the adversary must prove the knowledge of the corresponding private key. They proved by an attack against verifiable encrypted signatures OFE based on Waters signature [12] that security in the certified-key model does not imply security in the chosen-key model. Then, various OFE protocols are proposed [13–17], which most of them are secure in the multi-user setting and chosen-key model which we refer to some of them below.

Huang *et al.*'s proposed the notion of ambiguous OFE to prevent the verifier from misusing the signer's partial signature [13]. In an ambiguous OFE, the verifier cannot convince anybody except the arbitrator about the authorship of a partial signature generated by the signer. This notion enhanced in many works including [16] in which no outsider, including the arbitrator, can collect any evidence about the exchange between the signer and verifier even after a resolution by the arbitrator. Traditionally, the items being exchanged are digital signatures. Recently, OFE of ring signatures [14] (two members from two different groups can exchange their ring signatures in a fair way) and OFE of threshold signatures [15] (signatures are created by a subset of legitimate signers instead of a single signer) are considered. Wang *et al.* introduced attribute-based OFE which take into account user's attributes such as nationality and age [17].

In 2011, Huang *et al.* [6] formalized the notion of accountable OFE and provide a generic scheme, which enjoys the accountability along with the transparency while the scheme proposed in [8] has a non-transparent third party. They defined the security requirements needed for an accountable OFE and proved the security of their generic protocol in the random oracle model. In addition, they presented a concrete protocol as an example of their generic scheme.

Rivest *et al.* introduced the concept of ring signatures [18] which attract a great attention later [19–21]. Ring signatures allow a signer as a ring member, to sign a message on behalf of her ring anonymously, in order to identify her ring. An OR-signature can be viewed as a two-user ring signature. OR-signatures are created by applying the Fiat-Shamir heuristic [22] to the OR-proof [23]. Due to the application of the Fiat-Shamir heuristic, all OR-signatures are secure in the random oracle model.

1.2 Contributions

In this paper, we make the following contributions:

(1) We present the first generic OFE protocol with accountability property which is secure in the standard model. The partial signature is an ordinary signature. The full signature is a partial signature along with a traceable ring signature (TRS) in a ring which consists of the signer and the arbitrator. By using the traceability property of TRSs, we design a tracing algorithm to detect the dishonest party. We define a security model that is multi-user setting and chosen key model and prove the security of our protocol in this model.

(2) The partial signature in Huang *et al.*'s generic scheme [6] is an ordinary signature as well as our generic scheme. On the other hand, their full signature consists of a partial signature, a random number, an undeniable signature along with an OR-signature. The OR-signature which states that the producer of the undeniable signature is either the signer or arbitrator, is generated by using non-interactive zero-knowledge (NIZK) proofs. However, in our scheme the full signature is a partial signature along with a TRS. Thus, their full signature is more complex compared to our full signature in both generation and verification. Therefore, we obtain the accountability more efficiently. Huang *et al.* also presented a variant of their scheme when employing multiple arbitrators [6] that is more complex than our scheme since we use ring signatures rather than OR-signatures.

(3) Because of using the Fiat-Shamir heuristic [22] for generating OR-signatures, we cannot construct concrete protocols which are secure in the standard model from Huang *et al.*'s generic scheme [6]. However, in our generic scheme, we use ordinary signature schemes and TRS schemes which are proven secure in the standard model, therefore, it is possible to construct concrete OFE protocol without random oracle.

In Section 2, we define accountable OFE protocol and its security model. In Section 3, we review some preliminaries used in our scheme. In Section 4, we propose our generic protocol and prove its security in the standard model. Next, we present a concrete protocol which is secure in the random oracle model. Then, we provide a comparison with other OFE protocols. Finally, the paper is concluded in Section 7.

2 Syntax and Security Model

In this section, we define the accountable OFE and its security requirements in the multi-user setting and chosen-key model. The formal definitions of the following sections are deduced from those in [5, 6] with a small modification where needed.

2.1 Syntax of Accountable OFE

An accountable OFE protocol consists of the following algorithms [6]. The last three algorithms are additional algorithms in comparison with a traditional OFE protocol to make OFE accountable:

- $PMGen$: On input 1^k where k is the security parameter, it outputs the system parameter PM.
- $Setup^{TTP}$: On input PM, the algorithm outputs the arbitrator's public key PK_T and the corresponding secret key SK_T.
- $Setup^{User}$: On input PM and (optionally) PK_T, the algorithm outputs a public /secret key pair (PK, SK). We denote by (PK_i, SK_i) the user U_i key pair.
- $PSig$: On input (m, SK_i, PK_T), gives as output a signature σ_P, which is denoted as a *partial signature*.
- $PVer$: On input $(m, \sigma_P, PK_i, PK_T)$, gives as output accept ("1") or reject ("0").
- $FSig$: On input (m, SK_i, PK_T), gives as output a signature σ_F, which is denoted as a *full signature*.
- $FVer$: On input $(m, \sigma_F, PK_i, PK_T)$, gives as output accept ("1") or reject ("0").
- Res: The resolution algorithm takes as input $(m, \sigma_P, SK_T, PK_i)$ and gives as output a full signature σ_F, or $\perp$ showing the failure of the resolution.
- $Prove^{TTP}$: On input $(m, \sigma_F, PK_i, PK_T, SK_T)$, gives as output a proof π^{TTP} to claim or deny the signature.
- $Prove^{User}$: On input $(m, \sigma_F, PK_i, PK_T, SK_i)$, gives as output a proof π^{User} to claim or deny the signature.
- $Trace$: On input $(m, \sigma_F, PK_i, PK_T, \pi)$, gives as output PK_i, PK_T or $\perp$. Notice that for running the tracing algorithm we do not need any special secret. Thus, this algorithm could be run by anybody.

Correctness: If all the signatures are generated according to the protocol specification, we need the following to be true:

$$PVer(m, PSig(m, SK_i, PK_T), PK_i, PK_T) = 1$$
$$FVer(m, FSig(m, SK_i, PK_T), PK_i, PK_T) = 1$$
$$FVer(m, Res(m, PSig(m, SK_i, PK_T), SK_T, PK_i)$$
$$, PK_i, PK_T) = 1$$

In addition, if we have $(m, \sigma_F, PK_i, PK_T, \pi)$, where σ_F is a valid full signature on m and π is a proof generated by the $Prove^{User}$ or $Prove^{TTP}$ algorithm, we will have:

$$Trace(m, \sigma_F, PK_i, PK_T, \pi) = 1$$

Resolution ambiguity: Any full signature generated by the signer $FSig(m, SK_i, PK_T)$, is computationally indistinguishable from that resolved by the arbitrator $Res(m, PSig($
$m, SK_i, PK_T), SK_T, PK_i)$.

2.2 Accessible Oracles by Adversary

To formally define the security requirements of an accountable OFE, an adversary plays a game against a challenger and may have access to the following oracles:

- $\mathbf{O_{Res}}$ Refers to the resolution oracle that takes as input a partial signature σ_P of the user i on message m and outputs a full signature σ_F.
- $\mathbf{O_{PSig}}$ Refers to the partial signing oracle that takes as an input a message m and returns a partial signature σ_P under the PK_i.
- $\mathbf{O_{FSig}}$ Refers to the full signing oracle that takes as an input a message m and returns a full signature σ_F under the PK_i.
- $\mathbf{O_{Prove^{TTP(User)}}}$ On a valid message-signature pair (m, σ_F) under (PK_i, PK_j) as inputs, responds by a proof π.

2.3 Security Model

The security of an accountable OFE protocol consists of four aspects: accountability, security against signers, security against verifiers, and security against the arbitrator. The definitions of them in the multi-user setting and chosen-key model are as follows:

Accountability By an accountable OFE we mean that it provides the following definition of accountability:

(1) **Type I Accountability**: It is infeasible for the signer to produce a full signature which can be proved as an output of the Res algorithm.
(2) **Type II Accountability**: It is infeasible for the arbitrator to produce a full signature which can be proved as an output of the $FSig$ algorithm.
(3) **Type III Accountability**: It is infeasible for the arbitrator and the signer to both claim or deny a valid full signature.

We formally define the following experiment for the type I accountability:

$$\begin{aligned}
PM &\leftarrow PMGen(1^k)\\
(PK_T, SK_T) &\leftarrow Setup^{TTP}(PM)\\
(PK^*) &\leftarrow A(PK_T)\\
(m, \sigma_F, \pi) &\leftarrow A^{O_{Prove^{TTP}}, O_{Res}}(PK^*, PK_T)\\
\text{Success of } A &:= [FVer(m, \sigma_F, PK^*, PK_T) = 1 \wedge\\
&\quad Trace(m, \sigma_F, PK^*, PK_T, \pi)\\
&= PK_T \wedge (m, ., PK^*) \notin Query(A, O_{Res})]
\end{aligned}$$

PK^* is the chosen public key by the adversary who may not know the corresponding private key. The list $Query(A, O_{Res})$ refers to resolution queries that issued by A to the resolution oracle for resolving any partial signature with respect to the chosen public key without knowing the corresponding private key.

Definition 1. An OFE scheme is (t, q, ϵ)-type I accountable, if any PPT adversary that runs at most in time t, makes at most q queries, succeeds in the above experiment at most with ϵ probability.

For formal definition of the type II and III accountability, the reader can refer to [6].

Security Against Signers No PPT adversary can produce a partial signature which passes the $PVer$ algorithm, but cannot be converted to the full signature by the arbitrator with non-negligible probability. This aspect of security gives an assurance of fairness to the verifiers, i.e. the verifier always acquires the full signature from the arbitrator, if he has a partial signature from the signer. We formally define the following experiment:

$$\begin{aligned}
PM &\leftarrow PMGen(1^k)\\
(PK_T, SK_T) &\leftarrow Setup^{TTP}(PM)\\
(m, \sigma_P, PK^*) &\leftarrow A^{O_{Res}}(PK_T)\\
\sigma_F &\leftarrow Res(m, \sigma_P, SK_T, PK^*)\\
\text{Success of } \ A &:= [PVer(m, \sigma_P, PK^*, PK_T) = 1\\
&\wedge \ FVer(m, \sigma_F, PK^*, PK_T) = 0
\end{aligned}$$

In this experiment, the adversary can choose arbitrarily the public key PK^* without knowing the corresponding private key.

Definition 2. An OFE scheme is (t, q_{Res}, ϵ)-secure against signers, if any PPT adversary that runs at most in time t and makes at most q_{Res} resolution queries, succeeds in the above experiment at most with probability ϵ.

Security Against Verifiers No PPT adversary can generate a full signature without asking the signer or the arbitrator to produce it with non-negligible probability. We formally define the following experiment:

$$\begin{aligned}
PM &\leftarrow PMGen(1^k)\\
(PK_T, SK_T) &\leftarrow Setup^{TTP}(PM)\\
(PK, SK) &\leftarrow Setup^{User}(PM)\\
(m, \sigma_F) &\leftarrow A^{O_{PSig}, O_{Res}}(PK_T, PK)\\
\text{Success of } \ A &:= [FVer(m, \sigma_F, PK, PK_T) = 1\\
&\wedge (m, ., PK) \notin Query(A, O_{Res})
\end{aligned}$$

The verifier does not need to access the full signature oracle O_{FSig}, since she can produce the full signature by the resolution oracle O_{Res}, and the partial signature oracle O_{PSig}.

Definition 3. An OFE scheme is $(t, q_{Res}, q_{PSig}, \epsilon)$-secure against verifiers, if any PPT adversary that runs at most in time t, makes at most q_{Res} resolution queries and q_{PSig} partial signing queries, succeeds in the above experiment at most with probability ϵ.

Security Against the arbitrator No PPT adversary can generate a full signature without having the corresponding partial signature from the signer with non-negligible probability. This aspect of security gives an assurance of fairness to the signer that is, no one can frame the full signature without asking the signer to produce the partial signature. We formally define the following experiment:

$$\begin{aligned}
PM &\leftarrow PMGen(1^k)\\
(PK, SK) &\leftarrow Setup^{User}(PM)\\
(PK_T, SK_T^*) &\leftarrow A(PK)\\
(m, \sigma_F) &\leftarrow A^{O_{PSig}}(SK_T^*, PK_T, PK)\\
\text{Success of } \ A &:= [FVer(m, \sigma_F, PK, PK_T) = 1\\
&\wedge (m, .) \notin Query(A, O_{PSig})]
\end{aligned}$$

Where SK_T^* is A's state information, which might not be the corresponding private key of PK_T and the $Query(A, O_{PSig})$ refers to the partial signing queries that issued by the partial signing oracle.

Definition 4. An OFE scheme is (t, q_{PSig}, ϵ)-secure against the arbitrator, if any PPT adversary that runs at most in time t and makes at most q_{PSig} partial signing queries, succeeds in the above experiment at most with probability ϵ.

Definition 5. An accountable OFE scheme is said to be secure in the multi-user setting and chosen-key model if it is accountable, secure against signers, secure against verifiers and secure against the arbitrator.

3 Preliminaries

Our scheme is based on two primitives: an ordinary signature and a TRS.

3.1 Ordinary Signature

An ordinary digital signature scheme OS consists of three algorithms defined as below:

- $OS.KG$: On input 1^k, where k is the security parameter, outputs a key pair (pk, sk).
- $OS.Sig$: On input (m, sk), outputs a signature σ.
- $OS.Ver$: On input (m, σ, pk), outputs accept "1" or reject "0".

Correctness: We require that for every $k \in \mathbb{N}$, every (pk, sk) output by $OS.KG(1^k)$, and every message m in the appropriate underlying plaintext space, The following equation holds

$$OS.Ver(m, OS.Sig(m, sk), pk) = 1$$

The standard unforgeability game for an ordinary signature scheme is defined as follows [24]:

$$\begin{aligned}
(pk, sk) &\leftarrow OS.KG(1^k)\\
(m, \sigma) &\leftarrow A^{O_{OS.Sig}}(pk)\\
\text{Success of } \ A &:= [OS.Ver(m, \sigma, pk) = 1\\
&\wedge (m, .) \notin Query(A, O_{OS.Sig})]
\end{aligned}$$

The Adversary A is a PPT adversary and $O_{OS.Sig}$ is the signing oracle which takes as input a message

m and outputs a signature on m under pk. The list $Query(A, O_{OS.Sig})$ is the signing queries issued by A.

Definition 6. A signature scheme is said to be (t, q, ϵ)-unforgeable, if any PPT adversary that runs at most in time t and makes at most q signing queries, wins the experiment with at most ϵ probability.

3.2 Traceable Ring Signature

The ring signature allows a signer to sign a message on behalf of a ring anonymously and compared to group signatures does not need any group manager and special setup. However, it is vulnerable to malicious signers because of its anonymity. The traceable ring signature (TRS) restricts the anonymity of the signer [20]. The TRS has a tag that consists of a list of public keys and an issue which refers to a social event. If the signer signs a message with respect to the tag for the first time, she remains anonymous, but if she signs another message with respect to the same tag, the identity of the signer is revealed. A TRS consists of the following algorithms [20]:

- $TRS.KG$: Takes as input a security parameter k and outputs a key pair (pk, sk).
- $TRS.Sig$: Takes as input a secrect key sk_i, a tag $L = (issue, pk_N)$, a message m, and outputs σ, where $N = \{1, ..., n\}$ denotes an ordered list of signers and $PK_N = \{pk_1, ..., pk_n\}$ is a public key list.
- $TRS.Ver$: Takes as input a tag $L = (issue, pk_N)$, a message m, and a signature σ, and outputs "1" or "0".
- $TRS.Trace$: Takes as input a tag $L = (issue, pk_N)$, and two message-signature pairs, $(m, \sigma), (m', \sigma')$, and outputs "linked", "indep" or pk where $pk \in pk_N$.

Correctness: For every $k \in \mathbb{N}$, every $n \in \mathbb{N}$, every $i \in N = \{1, ..., n\}$, and every $issue \in \{0,1\}^*$, and every $m \in \{0,1\}^*$, if $(pk_N, sk_N) \leftarrow TRS.KG(1^k)$ and $\sigma \leftarrow TRS.Sig(m, L, sk_i)$, where $L = (issue, pk_N)$, we need the following to be true:

$$TRS.Ver(m, L, \sigma) = 1$$

3.2.1 Security Properties

The security requirements for a TRS are as follows [20]:

- *Traceability*: If a ring member signs two different messages with respect to the same tag, the identity of the signer can be traced.
- *Tag-linkabiity (One-more unforgeability)*: If a ring member signs twice with respect to the same tag, two signatures can be detected as linked.
- *Anonymity*: As long as a ring member signs once per tag, the identity of the signer remains anonymous.
- *Exculpability*: A honest ring member cannot be accused of signing twice with respect to the same tag. We formally define the following experiment for the exculpability:

$$\begin{aligned} (pk, sk) &\leftarrow TRS.KG(1^k) \\ (L', m, \sigma), (L', m', \sigma') &\leftarrow A^{O_{TRS.Sig_{sk}}}(pk) \\ \text{Success of } \ A &:= [TRS.Trace(L', m, \sigma, m', \sigma') = pk \\ &\wedge \textit{at least one of } (L', m) \textit{ or } (L', m') \\ &\notin Query(A, O_{TRS.Sig_{sk}}) \wedge pk \in L'] \end{aligned}$$

where $O_{TRS.Sig_{sk}}$ is the TRS oracle of the target user which takes as input a message m, a list of public keys where $pk \in L$, and outputs a TRS σ on m under the ring L. The list $Query(A, O_{TRS.Sig_{sk}})$ is the TRS queries issued by A.

Definition 7. A TRS scheme is (t, q, ϵ)-exculpable, if any PPT adversary that runs at most in time t, makes at most q queries, succeeds in the above experiment at most with probability ϵ.

- *Unforgeability*: The standard unforgeability game for a TRS scheme is defined as follows:

$$\begin{aligned} (pk_i, sk_i) &\leftarrow TRS.KG(1^k) \ \textit{for } i \in N = \{1, ..., n\} \\ T &:= (issue, PK_R) \\ (L', m, \sigma) &\leftarrow A^{O_{TRS.Sig}}(L) \\ \text{Success of } \ A &:= [TRS.Ver(m, \sigma, L') = 1 \\ &\wedge (., m, L') \notin Query(A, O_{TRS.Sig})] \end{aligned}$$

where oracle $O_{TRS.Sig}$ is the TRS oracle which takes as input a message m, an index i, a list of public keys L and outputs a TRS σ on m under the ring L.

Definition 8. A TRS scheme is said to be (t, q, ϵ)-unforgeable, if any PPT adversary that runs at most in time t and makes at most q TRS queries, wins the experiment with at most ϵ probability.

For formal definition of the other security requirements, the interested reader can refer to [20].

4 Our Generic Scheme and its Security Analysis

In this section, we give a detail description of our generic OFE protocol and also provide a security analysis of the proposed protocol.

4.1 Generic Scheme

There are many ordinary signatures and TRS schemes proven secure in the standard model such as [25], [21]. Using these schemes in our generic OFE protocol, it is possible to construct concrete and efficient

OFE protocols which are secure in the standard model. The partial signature is an ordinary signature with signer's secret key and the full signature constitutes of the partial signature along with a TRS in a ring consisting of the signer and the arbitrator. Let $OS = (OS.KG, OS.Sig, OS.Ver)$ be an ordinary signature scheme and $TRS = (TRS.KG, TRS.Sig, TRS.Ver, TRS.Trace)$ be a TRS scheme. Our accountable OFE protocol consists of the following algorithms:

- $PMGen$: On input the security parameter 1^k, this algorithm generates the parameters needed for the OS and TRS, which will be used in the other algorithms of OFE.
- $Setup^{TTP}$: The arbitrator computes (PK_T, SK_T) by running $(sk_T, pk_T) \leftarrow TRS.KG(1^k)$.
- $Setup^{User}$: Each user runs $(sk_i^1, pk_i^1) \leftarrow OS.KG(1^k)$ and $(sk_i^2, pk_i^2) \leftarrow TRS.KG(1^k)$ to compute the key pair $(SK_i, PK_i) = ((sk_i^1, sk_i^2), (pk_i^1, pk_i^2))$.
- $PSig$: To partially sign a message m, the signer computes an ordinary signature by running $OS.Sig(m, sk_i^1)$, where σ_P is a partial signature.
- $PVer$: To verify the partial signature σ_P on a message m, the verifier checks if $OS.Ver(m, \sigma_P, pk_i^1) = 1$.
- $FSig$: To fully sign a message m, the signer first computes the partial signature σ_P by running $OS.Sig(m, sk_i^1)$ and then computes a TRS by running $\sigma^{TRS} \leftarrow TRS.Sig(m \parallel \sigma_P \parallel PK_i, sk_i^2, L)$. The full signature is (σ_P, σ^{TRS}).
- $FVer$: To verify the signature σ_F on input a message m, the verifier first runs $OS.Ver(m, \sigma_P, pk_i^1)$ to verify σ_p and then runs $TRS.Ver(m \parallel \sigma_P \parallel PK_i, \sigma^{TRS}, L)$ to verify σ^{TRS}. If both the verification output 1, then accepts ; otherwise, rejects.
- Res: After receiving σ_P from the verifier, that is claimed to be generated by the signer, the arbitrator first checks the validity of σ_P by running $OS.Ver(m, \sigma_P, pk_i^1)$ and checks if the verifier did his job. Then he transfers the partial signature to the full signature σ_F by running $TRS.Sig(m \parallel \sigma_P \parallel PK_i, sk_T, L)$ and sets $\sigma_F = (\sigma_P, \sigma^{TRS})$.
- $Prove^{TTP}$: To claim or deny a signature $\sigma_F = (\sigma_P, \sigma^{TRS})$ on message m with respect to the tag L, the arbitrator runs $TRS.Sig(m' \parallel \sigma_P \parallel PK_i, sk_T, L)$ and computes $\sigma^{TRS'}$, where m' is an arbitrary message different from m and returns $\pi^{TTP} = (m', \sigma^{TRS'})$ as a proof.
- $Prove^{User}$: To claim or deny a signature $\sigma_F = (\sigma_P, \sigma^{TRS})$ on message m with respect to the tag L, the user runs $TRS.Sig(m'' \parallel \sigma_P \parallel PK_i, sk_i^2, L)$ and computes $\sigma^{TRS''}$, where m'' is an arbitrary message different from m and returns $\pi^{User} = (m'', \sigma^{TRS''})$ as a proof.
- $Trace$: To trace the producer of the message-signature pair (m, σ_F), where $\sigma_F = (\sigma_P, \sigma^{TRS})$, the algorithm proceeds as follows:
  (1) Verifies the validity of $\pi^{TTP} = (m', \sigma^{TRS'})$ by running $TRS.Ver(m' \parallel \sigma_P \parallel PK_i, \sigma^{TRS'}, L)$ and that $m \neq m'$. If either of the verification fails, the judgment will be done according to π^{User}; otherwise, it checks the relation between (m, σ^{TRS}) and $\pi^{TTP} = (m', \sigma^{TRS'})$ with respect to the tag L, by running $pk_i \leftarrow TRS.Trace((m \parallel \sigma_P \parallel PK_i, \sigma^{TRS}), (m' \parallel \sigma_P \parallel PK_i, \sigma^{TRS'}), L)$.
  (2) Verifies the validity of $\pi^{User} = (m'', \sigma^{TRS''})$ by running $TRS.Ver(m'' \parallel \sigma_P \parallel PK_i, \sigma^{TRS''}, L)$ and that $m \neq m''$. If either of the verification fails, the judgment will be done according to the π^{TTP}; otherwise, it checks the relation between (m, σ^{TRS}) and $\pi^{User} = (m'', \sigma^{TRS''})$ with respect to the tag L by running $TRS.Trace((m \parallel \sigma_P \parallel PK_i, \sigma^{TRS}), (m'' \parallel \sigma_P \parallel PK_i, \sigma^{TRS''}), L)$.
  (3) If the output of both 1 and 2 are the same, output $pk_i(PK_i \text{ } or \text{ } PK_T)$; otherwise, $\perp$. If either of the sides refuses to present a proof, the judgment will be done according to the other one.

The correctness property of the proposed protocol is easy to follow and the resolution ambiguity follows directly from the anonymity property of TRSs.

4.2 Security Analysis

Lemma 1. *Our OFE protocol is (t, q, ϵ)-type* I *accountable, if the underlying TRS is (t, q, ϵ)-exculpable.*

Proof. Let A be a PPT adversary who (t, q, ϵ)-breaks the accountability, we build a PPT algorithm A' which runs A as a subroutine and (t, q, ϵ)-breaks the exculpability of TRSs.

On input the target public key $pk = PK_T$, A' feeds A with PK_T, outputs PK^* where PK^* is the signer's public key. For a resolution query (m, σ_P, PK^*) from A, A' asks his $O_{TRS.Sig_{sk_T}}$ oracle to generate a TRS σ^{TRS} on $(m \parallel \sigma_P \parallel PK^*)$ under the tag $L = (issue, (PK^*, PK_T))$ and sets $\sigma_F = (\sigma_P, \sigma^{TRS})$ as a resolved signature. For a $Prove^{TTP}$ query $(m, \sigma_F, PK^*, PK_T)$, A' asks his $O_{TRS.Sig_{sk_T}}$ oracle to sign $(m' \parallel \sigma_P \parallel PK^*)$ under the tag L where $m \neq m'$ and returns $\pi = (m', \sigma^{TRS'})$ as a proof.

At the end of the game, A outputs $(\bar{m}, \bar{\sigma_F}, \bar{\pi})$, where $\bar{\pi} = \pi, \bar{\sigma_F} = (\bar{\sigma_P}, \bar{\sigma}^{TRS}), Trace(\bar{m}, \bar{\sigma_F}, PK^*, PK_T, \bar{\pi}) = PK_T$ and $(\bar{m}, ., PK^*) \notin Query(A, O_{Res})$. Now, A' outputs $(L, \bar{m} \parallel \sigma_P \parallel PK^*, \bar{\sigma}^{TRS}), (L, m' \parallel \sigma_P \parallel PK^*, \sigma^{TRS'})$ where $L = (issue, (PK^*, PK_T))$ and wins the game since $TRS.Trace(L, \bar{m} \parallel \sigma_P \parallel PK^*, \bar{\sigma}^{TRS}, m' \parallel \sigma_P \parallel PK^*, \sigma^{TRS'}) = PK_T$ and

$(L, \bar{m} \parallel \sigma_P \parallel PK^*) \notin Query(A', O_{TRS.Sig_{sk_T}})$.

We have shown that there is an algorithm A', which can (t, q, ϵ)-break the exculpability of TRSs, if there is an adversary that can (t, q, ϵ)-break the accountability of the OFE scheme. This is in contrast to the exculpability of TRSs. Therefore, our OFE protocol is type I accountable.

The types II and III accountability can be proved similarly.

Lemma 2. *Our OFE protocol is unconditionally secure against signers.*

Proof. The security against signers follows unconditionally from the ability of the arbitrator to produce a TRS. Since having a valid partial signature σ_P on a message m with respect to the public key PK_i, the arbitrator could always convert σ_P to the full signature $\sigma_F = (\sigma_P, \sigma^{TRS})$, by generating a TRS on $(m \parallel \sigma_P \parallel PK_i)$ with respect to the tag L, using his secret key SK_T.

Lemma 3. *Our OFE protocol is $(t, q_{Res}, q_{PSig}, \epsilon)$-secure against verifiers if the underlying TRS is $(t + t_1 q_{PSig}, q_{Res}, \epsilon)$-unforgeable, where t_1 is the time cost to generate one partial signature in the proposed protocol.*

Proof. Let A be the PPT adversary that $(t, q_{Res}, q_{PSig}, \epsilon)$-breaks the security against verifiers, we build a PPT algorithm A' which runs A as a subroutine and $(t + t_1 q_{PSig}, q_{Res}, \epsilon)$-breaks the unforgeability of TRSs.

On input, two challenge public keys pk_0 and pk_1 which are the ring public keys of the signer and arbitrator, A' generates a key pair $(pk, sk) \leftarrow OS.KG(1^k)$, chooses a random bit $b \leftarrow \{0, 1\}$ and sets $PK_T = pk_b$ and $PK = (pk, pk_{1-b})$. Then A' runs A on input the (PK_A, PK) and answers the partial signing queries of A by computing $OS.Sig(m, sk)$ and the resolution queries by asking the TRS oracle to sign $(m \parallel \sigma_P \parallel PK)$ under the ring (pk_0, pk_1) and returns $\sigma_F = (\sigma_P, \sigma^{TRS})$ as a resolved signature.

At the end of the game, A outputs $(m', \sigma'_F = (\sigma'_P, \sigma^{TRS'}))$ as a forgery, where $FVer(m', \sigma'_F, PK, PK_T) = 1$ and $(m', ., PK) \notin Query(A, Res)$. Now A' output $(m' \parallel \sigma'_P \parallel PK, \sigma^{TRS'})$ as a valid TRS on $(m' \parallel \sigma'_P \parallel PK)$ under the ring (pk_0, pk_1), where $(m' \parallel \sigma'_P \parallel PK)$ is never issued by A' to the TRS oracle. The running time of A' is the same as A plus the time it takes to answer the partial signing queries, which we denoted by t_1. Thus the total running time of A' is $t + t_1 q_{PSig}$.

We have shown that there is an algorithm A' which can $(t + t_1 q_{PSig}, q_{Res}, \epsilon)$-break the unforgeability of TRSs, if there is an adversary who can $(t, q_{Res}, q_{PSig}, \epsilon)$-break the security against the verifiers. This is in contrast to the unforgeability of TRSs. Therefore, our OFE protocol is secure against verifiers.

Lemma 4. *Our OFE protocol is (t, q_{PSig}, ϵ)-secure against the arbitrator, if the underlying ordinary signature is (t, q_{PSig}, ϵ)-unforgeable.*

Proof. Let A be the PPT adversary that (t, q_{PSig}, ϵ)-breaks the security against the arbitrator, we build a PPT algorithm A' which runs A as a subroutine and (t, q_{PSig}, ϵ)-breaks the unforgeability of the ordinary signature.

On input the public key pk_1 of ordinary signature OS, A' runs $(sk_2, pk_2) \leftarrow TRS.KG(1^k)$, passes $PK = (pk_1, pk_2)$ as input to A who outputs (SK^*_T, PK_T), where PK_T is the arbitrator's public key. Then A asks the O_{PSig} oracle for partial signing queries which can be simulated by A' using the $O_{OS.Sig}$ oracle. When A asks the O_{PSig} oracle to sign a message m, A' passes m to $O_{OS.Sig}$ oracle which returns as a valid partial signature.

At the end of the game A outputs $(m', \sigma'_F = (\sigma'_P, \sigma^{TRS'}))$ as a forgery where $FVer(m', \sigma'_F, PK, PK_T) = 1$ and . Now A' outputs (m', σ'_P) as a valid signature on m', where m' never issued by A' to the signing oracle $O_{OS.Sig}$.

We have shown that there is an algorithm A' which can (t, q_{PSig}, ϵ)-break the unforgeability of ordinary signature, if there is an adversary who can (t, q_{PSig}, ϵ)-break the security against the arbitrator. This is in contrast to the unforgeability of ordinary signatures. Therefore, our OFE protocol is secure against the arbitrator.

Theorem 1. *Our OFE protocol is secure in the multi-user setting and chosen-key model.*

Proof. The proof follows directly from the Lamma 1,2,3,4.

5 Our Concrete Protocol

Our concrete protocol which is secure in the random oracle model is an example of our generic scheme. However, due to the lack of efficiency, we leave it to the future to present a concrete protocol which is secure in the standard model. The protocol is based on the Schnor's ordinary signature [26] and the Fujisaki's TRS [20]. We use the same notation as in [20] for generating and verifying σ^{TRS}. The tag $L = (issue, pk_N)$ refers to the ring between the signers and the arbitrator, where $issue \in \{0, 1\}^*$ and pk_N is a list consists of public keys of signers and arbitrator. Let $OS = (OS.KG, OS.Sig, OS.Ver)$ be the Schnor's signature scheme and $TRS = (TRS.KG, TRS.Sig, TRS.Ver, TRS.Trace)$ be the Fujisaki's TRS scheme. The protocol consists of the following algorithms:

- $PMGen$: On input the security parameter 1^k, outputs (G, q, g, H, H', H'') where G is a multiplicative group of prime order q with generator g and $H : \{0,1\}^* \to G$, $H' : \{0,1\}^* \to G$, and $H'' : \{0,1\}^* \to \mathbb{Z}_q$.
- $Setup^{TTP}$: The arbitrator runs $TRS.KG(1^k)$ to compute $(PK_T, SK_T) = (x_T, (g, y_T, G))$, where $x_T \leftarrow \mathbb{Z}_q$ and $y_T = g^{x_T}$.
- $Setup^{User}$: Each user runs $(sk_i^1, pk_i^1) = (x_i^1, g^{x_i^1}) \leftarrow OS.KG(1^k)$ and $(sk_2^1, pk_i^2) = (x_i^2, g^{x_i^2}) \leftarrow TRS.KG(1^k)$ to compute the key pair $(SK_i, PK_i) = ((sk_i^1, sk_i^2), (pk_i^1, pk_i^2))$, where $x_i^1, x_i^2 \leftarrow \mathbb{Z}_q$.
- $PSig$: To partially sign a message $m \in \{0,1\}^*$, the signer chooses $r \leftarrow \mathbb{Z}_q$ and computes $\sigma_P = (\alpha_1, \alpha_2)$, where $\alpha_1 = H'(m, g^r)$ and $\alpha_2 = r - \alpha_1 x_i^1$.
- $PVer$: To verify the partial signature σ_P on a message m the verifier outputs "1", if $\alpha_1 = H'(m, g^{\alpha_2} pk_i^{1\,\alpha_1})$, otherwise "0".
- $FSig$: To fully sign a message m, the signer first computes the partial signature σ_P and then runs $TRS.Sig(x_i^2, m \parallel \sigma_P \parallel PK_i, L)$ and computes σ^{TRS} as follows:

(1) Compute $h = H(L)$ and $\sigma_i = h^{x_i^2}$.
(2) Compute $A_0 = H'(L, m \parallel \sigma_P \parallel PK_i)$ and $A_1 = (\sigma_i / A_0)^{1/i}$.
(3) For all $i \neq j$ compute $\sigma_j = A_0 A_1^j \in G$ (every $(j, log_h(\sigma_j))$ is on the line between $(i, log_h(\sigma_i))$ and $(0, log_h(A_0))$.
(4) Generate signature (c_N, z_N) on $(L, m \parallel \sigma_P \parallel PK_i)$ based on a NIZK proof of knowledge for the language

$$\mathcal{L} = \{(L, h, \sigma_N) \mid i \in N : log_g(pk_i^2) = log_h(\sigma_i)\}$$

Where (c_N, z_N) computed as follows:
  - Choose $w_i \leftarrow z_q$ and set $a_i = g^{w_i}, b_i = h^{w_i} \in G$.
  - Choose $z_j, c_j \leftarrow z_q$ and set $a_j = g^{z_j} pk_i^{2c_j}, b_j = h^{z_j} \sigma_j^{c_j} \in G$ for every $j \neq i$.
  - Set $c = H''(L, A_0, A_1, a_N, b_N)$ where $a_N = (a_1, ..., a_n)$ and $b_N = (b_1, ..., b_n)$.
  - Set $c_i = c - \sum_{i \neq j} c_j \bmod q$ and $z_i = w_i - c_i x_i^2 \bmod q$, and return $c_N = (c_1, ..., c_n)$ And $z_N = (z_1, , z_n)$ as a proof of $\mathcal{L}$.

(5) Output $\sigma_p = (A_1, c_N, z_N)$.
The full signature is $\sigma_F = (\sigma_P, \sigma^{TRS})$.

- $FVer$: To verify the signature σ_F on input message m, the verifier first runs $PVer(m, \sigma_P, pk_i^1)$ to verify σ_P and then runs $TRS.Ver(m \parallel \sigma_P \parallel PK_i, L, \sigma^{TRS})$ and verify σ^{TRS} as follows:

(1) Separates L as $(issue, pk_N)$ where $pk_N = \{pk_1^2, ..., pk_n^2\}$, checks $g, A_1 \in G, c_i, z_i \in z_q$ and $y_i \in G$ for all $i \in N$. Sets $h = H(L)$ and $A_0 = H'(L, m \parallel \sigma_P \parallel PK_i)$, and computes $\sigma_i = A_0 A_1^i \in G$ for all $i \in N$.
(2) Computes $a_i = g^{z_i} pk_i^{2c_i}, b_i = h^{z_i} \sigma_i^{c_i}$ for all $i \in N$
(3) Checks that $H''(L, A_0, A_1, a_N, b_N) = \sum_i c_i \bmod q$.
If both the verification output "1", accept; otherwise, reject.

- Res: After receiving σ_P from the verifier, which is claimed to be generated by the signer, the arbitrator first checks the validity of σ_P by running $PVer(m, \sigma_P, pk_i^1)$ and checks if the verifier did his job and then transfers the partial signature to the full signature σ_F by running $TRS.Sig(x_T, m \parallel \sigma_P \parallel PK_i, L)$ and sets $\sigma_F = (\sigma_P, \sigma^{TRS})$.
- $Prove^{TTP}$: To claim or deny a signature pair $\sigma_F = (\sigma_P, \sigma^{TRS})$ on a message m with respect to the tag L, the arbitrator runs again $\sigma^{TRS'} \leftarrow TRS.Sig(x_T, m' \parallel \sigma_P \parallel PK_i, L)$ and computes $\sigma^{TRS'}$, where m' is an arbitrary message different from m.
- $Prove^{User}$: To claim or deny a signature pair $\sigma_F = (\sigma_P, \sigma^{TRS})$ on a message m with respect to the tag L, the user runs again $\sigma^{TRS''} \leftarrow TRS.Sig(x_i, m'' \parallel \sigma_P \parallel PK_i, L)$ and computes $\sigma^{TRS''}$, where m'' is an arbitrary message different from m.
- $Trace$: To trace the generator of the message-signature pair (m, σ_F), where $\sigma_F = (\sigma_P, \sigma^{TRS})$, the algorithm does as follows:

(1) Verifies the validity of $\pi^{TTP} = (m', \sigma^{TRS'})$ by running $TRS.Ver(m' \parallel \sigma_P \parallel PK_i, \sigma^{TRS'}, L)$ and that $m \neq m'$. If either of the verification fails, the judgment will be done according to π^{User}; otherwise, it checks the relation between (m, σ^{TRS}) and $\pi^{TTP} = (m', \sigma^{TRS'})$ with respect to the tag L, where $\sigma^{TRS} = (A_1, c_N, z_N)$ and $\sigma^{TRS'} = (A_1', c_N', z_N')$ as follows:
  - Separates L to $(issue, pk_N)$ and sets $h = H(L)$, $A_0 = H'(L, m \parallel \sigma_P \parallel PK_i)$ and compute $\sigma_i = A_0 A_1^i$ for all $i \in N$. Does the same thing for $\sigma^{TRS'}$ and computes $\sigma_i^{'}$ for all $i \in N$.
  - For all $i \in N$, if $\sigma_i = \sigma_i^{'}$, stores pk_i in $TList$, where $TList$ is empty initially.
  - Outputs PK_T, if $pk_i = y_T$ is the only entry in $TList$, and PK_i otherwise.

(2) Verifies the validity of $\pi^{User} = (m'', \sigma^{TRS''})$ by running $TRS.Ver(m'' \parallel \sigma_P \parallel PK_i, \sigma^{TRS''}, L)$ and that $m \neq m''$. If either of the verification fails, the judgment will be done according to π^{TTP}; otherwise, it checks the relation between (m, σ^{TRS}) and $\pi^{User} = (m'', \sigma^{TRS''})$ with respect to the tag L, where $\sigma^{TRS} = (A_1, c_N, z_N)$ and $\sigma^{TRS''} = (A_1'', c_N'', z_N'')$ as follows:
  - Separates L to $(issue, pk_N)$ and sets $h =$

$H(L)$, $A_0 = H'(L, m \parallel \sigma_P \parallel PK_i)$ and computes $\sigma_i = A_0 A_1^i$ for all $i \in N$. Does the same thing for $\sigma^{TRS''}$ and computes $\sigma_i^{''}$ for all $i \in N$.

- For all $i \in N$, if $\sigma_i = \sigma_i^{''}$, stores pk_i in $TList$, where $TList$ is empty initially.
- Outputs PK_i, if $pk_i = pk_i^2$ is the only entry in $TList$, and PK_T otherwise.

(3) If the output of both 1 and 2 are the same, outputs $pk_i(PK_i \text{ or } PK_T)$; otherwise, $\perp$. If either sides refuses to present a proof, the judgment will be done according to the other one.

The concrete protocol can be proven secure in the multi-user setting and chosen-key model under the random oracle model. The proofs are similar to those in the generic scheme and we just give the intuition behind them here.

Resolution ambiguity requires that the Fujisaki's TRS be anonymous which is reduced to the Decisional Diffie-Hellman (DDH) assumption in the random oracle model.

Accountability requires that the Fujitsu's TRS be exculpable which is reduced to the discrete logarithm assumption in the random oracle model.

Security against verifiers requires that the Fujisaki's TRS be unforgeable which is reduced to the discrete logarithm assumption in the random oracle model.

Security against the arbitrator requires that the schnor's signature be unforgeable which is reduced to the discrete logarithm assumption in the random oracle model.

6 Comparison

We note that the schemes proposed in [6, 8] have a common property as our scheme, i.e. accountability. In Table 1, we compare our scheme to these protocols as well as the scheme proposed in [5] which motivates our work. The comparison is made from the following properties: multi-user setting or single-user setting, chosen-key model or certified model, resolution ambiguity, accountability, setup-free or setup-driven, random oracle model or standard model.

7 Conclusion

In this paper, first we introduced different OFE constructions. Next, we investigated the security problem raised in OFE protocols due to the arbitrator-verifier collusion. We observed that the mentioned problem is addressed by the accountable OFE protocol. However, the existing protocol is secure in the random oracle model, therefore, we proposed the first generic accountable OFE protocol, which is secure in the standard model. Our protocol is based on ordinary signatures and TRS schemes. Since, there are ordinary signatures and TRS schemes secure in the standard model, it is possible for us to construct concrete accountable OFE protocols which are secure in the standard model.

Table 1. A brief comparison with some related protocols

Schemes	Multi-user	Chosen-key	Resolution ambiguity	Accounta-bility	Setup-free	Random oracle
[8]	N	N	N	Y	Y	Y
[5]	Y	Y	Y	N	Y	N
[6]	Y	Y	Y	Y	Y	Y
[Ours]	Y	Y	Y	Y	Y	N

We presented a concrete accountable OFE protocol which is secure in the random oracle model as an example of our generic scheme. However, due to the lack of efficiency, we leave it as our future work to construct a concrete accountable OFE protocol that is secure without random oracles.

References

[1] R.Ganjavi, M. Rajabzadeh Asaar and M. Salmasizadeh. A traceable optimistic fair exchange protocol. In 11th International ISC Conference on. Information Security and Cryptology (ISCISC), pages 161-166 , 2014. IEEE.

[2] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, Advances in Cryptology EUROCRYPT 2003, Lecture Notes in Computer Science 2656, pages 416432, 2003. Springer-Verlag.

[3] C. Boyd, Digital multisignatures, Cryptography and coding (1986).

[4] M. Park, E. Chong, H. Siegel, and I. Ray. Constructing fair exchange protocols for E-commerce via distributed computation of RSA signatures. In 22th Annual ACM Symp. on Principles of Distributed Computing, pages 172181, 2003.

[5] Q. Huang, G. Yang, Duncan S. Wong, and W. Susilo. Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In Proc. of The Cryptographers' Track at the RSA Conference-CT-RSA 2008, Lecture Notes in Computer Science 4964, pages 106-120, Berlin, 2008. Springer.

[6] X. Huang, Y. Mu, Y. Susilo, W. Wu, J. Zhou, and R. H. Deng. Preserving Transparency and Accountability in Optimistic Fair Exchange of Digital Signatures. IEEE Transaction on Information

Forensics and Security 6(2), pages 498512, 2011.

[7] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In Proc. of the 4th ACM conference on Computer and Communications Security, pages 7-17, New York, 1997. ACM.

[8] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures (Extended abstract). In Proc. of International Conference on the Theory and Application of Cryptographic Techniques-EUROCRYPT'98, Lecture Notes in Computer Science 1403, pages 591-606, Berlin, 1998. Springer.

[9] Y. Dodis and L. Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In Proc. of the 3rd ACM Workshop on Digital Rights Management, pages 47-54, New York, 2003. ACM.

[10] Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic fair exchange in a multi-user setting. In Proc. of the 10th International Conference on Practice and Theory in Public-Key Cryptography-PKC 2007, Lecture Notes in Computer Science 4450, pages 118-133, Berlin, 2007. Springer.

[11] X. Huang, Y. Mu, W. Susilo, W. Wu, Y. Xiang. Optimistic Fair Exchange with Strong Resolution-Ambiguity. IEEE Journal on Selected Areas in Communications 29(7), 2011.

[12] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters. Sequential aggregate signatures and multisignatures without random oracles. In Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques-EUROCRYPT 2006, Lecture Notes in Computer Science 4004, pages 465-485, Berlin, 2006. Springer.

[13] Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Ambiguous optimistic fair exchange. In Proc. of the 14th International Conference on the Theory and Application of Cryptology and Information Security-ASIACRYPT 2008, Lecture Notes in Computer Science 5350, pages 74-89, Berlin, 2008. Springer.

[14] Q. Lie, G. Wang, and Y. Mu. Optimistic fair exchange of ring signatures. In Security and Privacy in Communication Networks 2012, Lecture Notes in Computer Science 96, pages 227-242, Berlin, 2012. Springer.

[15] Y. Wang, M. H. Au, J. K. Liu, T. H. Yuen, and W. Susilo. Threshold-Oriented Optimistic Fair Exchange. In Network and System Security 2013, Lecture Notes in Computer Science 7873, pages 424-438, 2013. Springer.

[16] Q. Huang, D. S. Wong, and W. Susilo. P2OFE: Privacy-Preserving Optimistic Fair Exchange of Digital Signatures. In Topics in CryptologyCT-RSA 2014, Lecture Notes in Computer Science 8366, pages 367-384, 2014. Springer.

[17] Y. Wang, M. H. Au, and W. Susilo. Attribute-based optimistic fair exchange: How to restrict brokers with policies. Theoretical Computer Science 527, pages 83-96, 2014.

[18] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In Proc. Asiacrypt 2001, Lecture Notes in Computer Science 2248, pages 552565, 2001. Springer.

[19] H. Shacham and B. Waters. Efficient ring signatures without random oracles. In Proc. Public-Key Cryptography-PKC 2007, Lecture Notes in Computer Science 4450, pages 166180, 2007. Springer.

[20] E. Fujisaki, K. Suzuki. Traceable ring signature. In Proc. Public-Key Cryptography-PKC 2007, Lecture Notes in Computer Science 4450, pages 181-200, 2007. Springer.

[21] E. Fujisaki. Sub-linear Size Traceable Ring Signatures without Random Oracles. In Cryptographers' Track at the RSA Conference-CT-RSA 2011, Lecture Notes in Computer Science 6558, pages 393-415, 2011. Springer.

[22] A. Fiat and A. Shamir. How to prove yourself. In Proc. Crypto 1986, Lecture Notes in Computer Science 263, pages 186-194, 1986. Springer.

[23] R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of partial knowledge and simplied design of witness hiding protocols. In Proc. Crypto 1994, Lecture Notes in Computer Science 839, pages 174-187, 1994. Springer.

[24] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attack. In SIAM J. Computing 17(2), pages 281308, 1988.

[25] D. Boneh and X. Boyen. Short signatures without random oracles. In Proc. Eurocrypt 2004, Lecture Notes in Computer Science 3027, pages 5673, 2004. Springer.

[26] C. P. Schnorr, Efficient identification and signatures for smart cards, in Proc. Crypto89, 1990, vol. 435, pp. 239252, LNCS, Springer.

Ramin Ganjavi received his B.S. degree in Electrical Engineering from Shahid Bahonar University of Kerman, Kerman, Iran, in 2012, and received his M.S. degree in Electrical Engineering from Sharif University of Technology, Tehran, Iran in 2014. His research interests include design and analysis of cryptographic protocols, cloud and big data security, security and privacy in Internet of Things.

Maryam Rajabzadeh Asaar received her B.S. degree in Electrical Engineering from Shahid Bahonar University of Kerman, Kerman, Iran, in 2004, and received her M.S. and Ph.D. degrees in Electrical Engineering from Sharif University of Technology, Tehran, Iran in 2008 and 2014, respectively. Her research interests include provable security, digital signatures, design and analysis of cryptographic protocols and network security.

Mahmoud Salmasizadeh received the B.S. and M.S. degrees in electrical engineering from Sharif University of Technology, Tehran, Iran, in 1972 and 1989, respectively. He also received the Ph.D. degree in information technology from Queensland University of Technology, Australia, in 1997. Currently, he is an associate professor in the Electronics Research Institute and adjunct associate professor in the Electrical Engineering Department, Sharif University of Technology. His research interests include design and cryptanalysis of cryptographic algorithms and protocols, E-commerce security, and information theoretic secrecy. He is a founding member of Iranian Society of Cryptology.