

University of Wollongong
Research Online

Faculty of Engineering and Information
Sciences - Papers: Part A

Faculty of Engineering and Information
Sciences

1-1-2014


P2OFE: Privacy-preserving optimistic fair exchange of digital signatures

Qiong Huang
South China Agricultural University

Duncan S. Wong
City University of Hong Kong, dwong@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Follow this and additional works at: <https://ro.uow.edu.au/eispapers>

 Part of the [Engineering Commons](#), and the [Science and Technology Studies Commons](#)

Recommended Citation

Huang, Qiong; Wong, Duncan S.; and Susilo, Willy, "P2OFE: Privacy-preserving optimistic fair exchange of digital signatures" (2014). *Faculty of Engineering and Information Sciences - Papers: Part A*. 1921.
<https://ro.uow.edu.au/eispapers/1921>

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

P2OFE: Privacy-preserving optimistic fair exchange of digital signatures

Abstract

How to sign an electronic contract online between two parties (say Alice and Bob) in a fair manner is an interesting problem, and has been studied for a long time. Optimistic Fair Exchange (OFE) is an efficient solution to this problem, in which a semi-trusted third party named arbitrator is called in to resolve a dispute if there is one during an exchange between Alice and Bob. Recently, several extensions of OFE, such as Ambiguous OFE (AOFE) and Perfect AOFE (PAOFE), have been proposed to protect the privacy of the exchanging parties. These variants prevent any outsider including the arbitrator from telling which parties are involved in the exchange of signatures before the exchange completes. However, in PAOFE, AOFE, and all the current work on OFE, the arbitrator can always learn the signer's signature at (or before) the end of a resolution, which is undesirable in some important applications, for example, signing a contract between two parties which do not wish others to find out even when there is a dispute that needs a resolution by the arbitrator. In this work, we introduce a new notion called Privacy- Preserving Optimistic Fair Exchange (P2OFE), in which other than Alice and Bob, no one else, including the arbitrator, can collect any evidence about an exchange between them even after the resolution of a dispute. We formally define P2OFE and propose a security model. We also propose a concrete and efficient construction of P2OFE, and prove its security based on the Strong Diffie-Hellman and Decision Linear assumptions in the standard model.

Keywords

digital, exchange, fair, p2ofe, signatures, preserving, optimistic, privacy

Disciplines

Engineering | Science and Technology Studies

Publication Details

Huang, Q., Wong, D. S. & Susilo, W. (2014). P2OFE: Privacy-Preserving Optimistic Fair Exchange of Digital Signatures. *Lecture Notes in Computer Science*, 8366 (2014), 367-384.

P²OFE: Privacy-Preserving Optimistic Fair Exchange of Digital Signatures

Qiong Huang¹, Duncan S. Wong², and Willy Susilo³

¹ South China Agricultural University, Guangzhou 510642, China.

² City University of Hong Kong, Hong Kong S.A.R., China.

³ University of Wollongong, Wollongong, NSW 2522, Australia.

csqhuang@alumni.cityu.edu.hk, duncan@cityu.edu.hk, wsusilo@uow.edu.au

Abstract. How to sign an electronic contract online between two parties (say Alice and Bob) in a fair manner is an interesting problem, and has been studied for a long time. Optimistic Fair Exchange (OFE) is an efficient solution to this problem, in which a semi-trusted third party named *arbitrator* is called in to resolve a dispute if there is one during an exchange between Alice and Bob. Recently, several extensions of OFE, such as Ambiguous OFE (AOFE) and Perfect AOFE (PAOFE), have been proposed to protect the privacy of the exchanging parties. These variants prevent any outsider including the arbitrator from telling which parties are involved in the exchange of signatures before the exchange completes.

However, in PAOFE, AOFE, and all the current work on OFE, the arbitrator can always learn the signer's signature at (or before) the end of a resolution, which is undesirable in some important applications, for example, signing a contract between two parties which do not wish others to find out even when there is a dispute that needs a resolution by the arbitrator. In this work, we introduce a new notion called *Privacy-Preserving Optimistic Fair Exchange* (P²OFE), in which other than Alice and Bob, no one else, including the arbitrator, can collect any evidence about an exchange between them even after the resolution of a dispute. We formally define P²OFE and propose a security model. We also propose a concrete and efficient construction of P²OFE, and prove its security based on the Strong Diffie-Hellman and Decision Linear assumptions in the standard model.

Keywords. optimistic fair exchange; signature; ambiguity; privacy preserving

1 Introduction

The fair exchange problem is about constructing a protocol for two parties, Alice and Bob, that allow them to exchange items in an all-or-nothing (fair) manner, that is, after the protocol, either both parties obtain the

other’s item or none of them does. There are two major approaches to do fair exchange. The first one is to have the parties release their secrets ‘gradually’, e.g. bit by bit, in multiple rounds. Besides, it is assumed that both of them have comparable computation power. Thus, this approach may not be appropriate for practical use.

Another approach is to have a third party called *arbitrator* employed. The arbitrator is semi-trusted by the two parties, and is usually offline. The arbitrator only gets involved when there is a dispute. Asokan *et al.* proposed this notion called *Optimistic Fair Exchange* (OFE) [1], and later extended it to support the exchange of digital signatures [2]. In OFE, Alice prepares an ‘encapsulated’ version of her signature, called *partial signature* σ_A , and sends it to Bob. If σ_A is valid, Bob returns his *full signature* ζ_B to Alice. In the third move, Alice tells Bob how to open σ_A or directly sends her full signature ζ_A to Bob if she believes ζ_B is valid. Figure 1 shows a normal execution. If Alice refuses or fails to return ζ_A , Bob resorts to the arbitrator for resolving σ_A . After checking the fulfillment of Bob’s obligation, the arbitrator extracts ζ_A from σ_A , and sends it to Bob. Figure 2 shows the case in which there is a dispute.

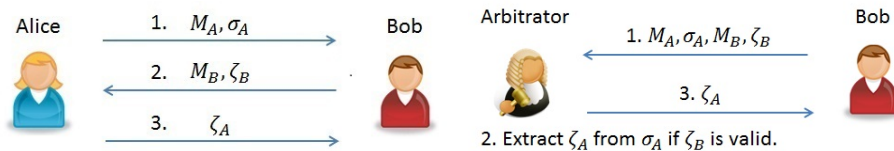


Fig. 1. OFE: Normal Execution

Fig. 2. OFE: Resolution

Due to the simple and elegant framework, and the low level of trust required on the third party, OFE has many useful applications. One of them is to sign contracts between two online parties. For example, Alice wants to buy a software from Bob’s online shop. She generates a partial signature on a message “Bob can withdraw \$100 from my bank account”. Bob then gives Alice his full signature on message “Alice can get a copy of Windows 13 from my shop”. If everything goes well, Alice gets the software and Bob gets the money from Alice’s bank account. If Bob does not get the full signature from Alice subsequently, Bob asks the arbitrator for resolving Alice’s partial signature and gets Alice’s full signature.

(*On the Privacy of OFE and its Variants*). In conventional OFE, Alice’s partial signature σ_A already reveals her will/intention to do exchange with Bob, from which Bob may take advantage of, and could be unfair

to Alice. In [13, 19], the notion of *Ambiguous Optimistic Fair Exchange* (AOFE)⁴ was introduced to solve this problem. In AOFE, Bob is endowed with the ability of producing partial signatures computationally indistinguishable from those of Alice. Recently, Wang *et al.* [30] proposed an enhanced version of AOFE, named *Perfect AOFE* (PAOFE), in which a partial signature leaks no information about the actual signer or the intended verifier. This is useful for applications where the involved parties of an exchange wish to further protect their privacy on whether they are indeed involved in an exchange or not. For instance, when Alice and Bob sign a business contract (e.g. a procurement deal) online while revealing and confirming who is involved in the process may potentially be harmful to (for example, the image of) Alice and/or Bob. No one including the arbitrator can tell who and what exchange has taken place from the transcript of a normal execution of PAOFE.

Although the privacy is ensured in a normal execution of PAOFE, this is not the case if a dispute occurs and a resolution is solicited. At the end of a resolution protocol run in PAOFE, the arbitrator gets the full signature ζ_A of Alice. (Note that the resolution is an algorithm run by the arbitrator in (A)OFE while it is a protocol in PAOFE.) Hence the arbitrator can confirm whether a particular party, say Alice, is involved in an exchange of signatures. Note that this is always the case in (A)OFE as the resolution algorithm outputs ζ_A . Whereas there are applications in which the parties do not want anyone including the arbitrator to confirm and especially, convince others their involvement even when there is a dispute that needs the arbitrator to resolve. Even in the example above, revealing and confirming who is involved in the business contract to the arbitrator during a dispute may potentially hurt (the image of) Alice and/or Bob. We stress here that revealing the contract (i.e. the message) itself (*without the signatures*) does not entail any concern on revealing, or letting outsiders or the arbitrator to confirm the involvement of a particular party in an exchange. This is because such a contract/message can be made up by anyone. Only the signed contract can be used to confirm a party's involvement. In this scenario, PAOFE would not help because the arbitrator learns the final signature ζ_A at the end of the resolution and hence can confirm the involvement of Alice. The arbitrator can even convince others about Alice's involvement by making use of ζ_A .

⁴ It is named *abuse-free contract signing* in [13] and *ambiguous optimistic fair exchange* in [19]. Hereafter we call it ambiguous optimistic fair exchange (AOFE), for the sake of the ease of presentation.

Our Contributions. In this paper we contribute to the study of fair exchange in the following aspects:

1. We introduce the notion of *Privacy-Preserving* OFE (P²OFE). The new notion differs from PAOFE mainly in that P²OFE explicitly requires that even the arbitrator cannot learn the signer’s full signature. The resolution in P²OFE is a protocol between the verifier and the arbitrator, and consists of two algorithms, Res^A and Res^V. Briefly, After receiving a partial signature σ for resolution, the arbitrator runs Res^A to convert it to an intermediate value θ , and gives to the verifier, who then runs Res^V to extract the signer’s full signature ζ from θ . It is required that without the intended verifier’s secret key, anyone cannot recover ζ from the intermediate value.
2. We present the security models of P²OFE to capture our intuition that even the arbitrator is unable to recover the signer’s full signature after the resolution. As in [16, 18] we consider the certified-key model in this paper, which is slightly weaker than the chosen-key model considered in [20, 30]. However, the perfect ambiguity in our model is stronger in the sense that we allow the adversary to interact with the intended verifier for resolution, which is not allowed in [30].
3. We also propose a concrete and efficient P²OFE protocol, the security of which is based on Strong Diffie-Hellman assumption [7] and Decision Linear assumption [8] without random oracles. Roughly, our protocol follows the sign-then-encrypt paradigm (which is common in the construction of designated confirmer signatures [9, 18]). A full signature is simply a Boneh-Boyen short signature [7], while a partial signature is a ‘twisted’ double encryption of the full signature. Please refer to Sec. 5 for the detailed construction.

2 Related Works

Since the introduction, OFE has attracted a lot of attention, e.g. [3, 10, 11, 15–22, 28, 29]. In [10], Dodis *et al.* showed a gap between the security of OFE in single-user setting (where there are one signer and one verifier) and that in multi-user setting (where there are multiple signers and verifiers). Using random oracle heuristic, they proposed a OFE secure in the multi-user setting and registered-key model [5]. Huang *et al.* [21] proposed a generic construction of OFE from time capsule signature [12], based on their observation on the similarity between the two primitives. The resulting protocol is secure in the multi-user setting and certified-key model without random oracles. Huang *et al.* [20] further strengthened Dodis *et*

al.'s result by relaxing the restriction on using a public key. They demonstrated that there is a gap between the security of OFE in chosen-key model [27] (in which an adversary can use any public key) and that in registered-key model. A generic construction using a standard signature and a two-user ring signature was also proposed and proven secure in the multi-user setting and chosen-key model.

In traditional OFE, Alice's partial signature is generally self-authenticating and indicates her commitment to some message already. This may allow Bob to take advantage of it. Garay *et al.* [13] and Huang *et al.* [19] addressed this problem and proposed notions of *abuse-free* OFE and *ambiguous* OFE, respectively. In both notions, Alice and Bob should be able to produce indistinguishable partial signatures so that given a valid partial signature from Alice, Bob cannot transfer the conviction to others. In this paper we universally call them as AOFE. Garay *et al.* constructed an efficient AOFE from a type of signatures called private contract signatures (PCS). Their PCS scheme is built from designated-verifier signature [23], and is secure in the registered-key model with random oracles. Huang *et al.* [19] proposed another efficient construction of AOFE using Groth-Sahai non-interactive proofs [14]. The scheme is secure based on Strong Diffie-Hellman assumption [7] and Decision Linear assumption [8] in the chosen-key model without random oracles. However, the scheme suffers from long signatures, which consist of more than 40 group elements.

Huang *et al.* [15, 16] proposed a new approach to constructing *interactive* AOFE, in which the signer interacts with the verifier to produce the partial signature. Their construction applies to a specific class of designated confirmer signature (DCS) [9] schemes, in which anyone is able to sample confirmer signatures from the signer's signature space efficiently, e.g. in polynomial time. However, not many DCS schemes enjoy this property, and thus limiting the application of Huang *et al.*'s construction. The authors improved the result by proposing another construction of AOFE from standard DCS [18]. They also proposed an efficient DCS construction, which follows the sign-then-encrypt paradigm. By applying their construction, they obtained an AOFE protocol which has short partial signature and the shortest full signature, and is secure based on SDH and DLIN assumptions without random oracles.

Huang *et al.* also introduced another variant of AOFE, called *group-oriented optimistic fair exchange* (GOFE) [17]. In GOFE, two users exchange signatures on behalf of their respective groups in a fair and anonymous manner so that either each group receives the other group's signa-

ture or none of them does, and in the meanwhile the users' identities are kept secret to others except their respective group managers.

Wang *et al.* proposed the notion of *perfect ambiguous optimistic fair exchange* (PAOFE) [30], in which only the intended verifier is able to tell which parties are involved in the exchange. They proposed a generic PAOFE construction by combining an AOFE protocol and a public key encryption scheme with *key privacy* (no one is able to tell whom a ciphertext is intended for). However, no concrete implementation of PAOFE is provided in [30].

In terms of the arbitrator not learning the exchanged material even in case of a dispute, there are also some other works in the non-signature exchange fields. For example, Belenkiy *et al.* [6] and Küpçü *et al.* [25] studied the privacy in optimistic fair exchange of files, where the arbitrator could not learn the full files. Avoine *et al.* [4] proposed to distribute the arbitrator so that no single arbitrator may learn the full signature. Similar idea has been used in [26] in the exchange of files.

3 Privacy-Preserving OFE

3.1 Definition

A Privacy-Preserving Optimistic Fair Exchange protocol (P²OFE) ‘blinds’ the arbitrator so that the arbitrator is unable to recover a full signature. Similar to PAOFE, the resolution in the definition of P²OFE below is a protocol rather than an algorithm in a conventional (A)OFE.

Definition 1. *A Privacy-Preserving Optimistic Fair Exchange protocol (P²OFE) involves the users (signers and verifiers) and the arbitrator, and consists of the following probabilistic polynomial-time (p.p.t. for short) algorithms and protocols:*

PMG. *It takes 1^k as input where k is the security parameter and outputs the system parameter PM.*

Akg. *It takes as input PM and outputs a key pair for the arbitrator. We denote it by $(\text{Apk}, \text{Ask}) \leftarrow \text{Akg}(\text{PM})$.*

UKg. *It takes PM (and optionally Apk) as input and outputs a user key pair. We denote it by $(\text{Pk}, \text{Sk}) \leftarrow \text{Ukg}(\text{PM}, \text{Apk})$.*

PSig. *This is the partial signature generation algorithm. It takes as input a message M , the signer's secret key Sk_i , the signer's public key Pk_i , the verifier's public key Pk_j and the arbitrator's public key Apk , and outputs a partial signature σ . We denote it by $\sigma \leftarrow \text{PSig}(M, \text{Sk}_i, \text{Pk}_i, \text{Pk}_j, \text{Apk})$.*

PVer. *This is for verifying a partial signature. It can be either an algorithm or a protocol, depending on whether the verification requires the interaction between the signer U_i and the verifier U_j . If the verification is non-interactive, the algorithm takes as input $(M, \sigma, \text{Pk}_i, \text{Pk}_j, \text{Apk}, \text{Sk}_j)$ and outputs a bit b . We denote it by $b \leftarrow \text{PVer}(M, \sigma, \text{Pk}_i, \text{Pk}_j, \text{Apk}, \text{Sk}_j)$. In case the verification is an interactive protocol, the common input consists of $(M, \sigma, \text{Pk}_i, \text{Pk}_j, \text{Apk})$. The signer (acting as the prover) has private input Sk_i and the randomness r used in signature generation, while the verifier has private input Sk_j . We denote a run of the protocol by*

$$b \leftarrow \text{PVer}_{\langle U_i(\text{sk}_i, r), U_j(\text{sk}_j) \rangle}(M, \sigma, \text{Pk}_i, \text{Pk}_j, \text{Apk}),$$

where b is the decision bit of U_j , which is 1 for acceptance and 0 for rejection.

Sig. *This is the full signature generation algorithm. It takes as input $(M, \text{Sk}_i, \text{Pk}_i, \text{Pk}_j, \text{Apk})$ and outputs a full signature ζ . We denote it by $\zeta \leftarrow \text{Sig}(M, \text{Sk}_i, \text{Pk}_i, \text{Pk}_j, \text{Apk})$.*

Ver. *This is for verifying a full signature. It takes as input $(M, \zeta, \text{Pk}_i, \text{Pk}_j, \text{Apk})$ and outputs a bit b which is 1 if ζ is a valid full signature of Pk_i and 0 otherwise. We denote it by $b \leftarrow \text{Ver}(M, \zeta, \text{Pk}_i, \text{Pk}_j, \text{Apk})$.*

Res. *This is a protocol between verifier U_j and arbitrator A for converting a partial signature to a full one. It consists of two algorithms, Res^A and Res^V . Res^A is run by the arbitrator for resolving a partial signature. It takes as input $(M, \text{Ask}, \sigma, \text{Pk}_i, \text{Pk}_j)$, and outputs an intermediate signature θ or \perp indicating the failure of resolution. Res^V is run by the intended verifier for extracting the full signature ζ from an intermediate signature θ . It takes as input $(M, \text{Sk}_j, \theta, \text{Pk}_i, \text{Pk}_j, \text{Apk})$ and outputs a full signature ζ . We denote the two algorithms by $\theta \leftarrow \text{Res}^A(M, \text{Ask}, \sigma, \text{Pk}_i, \text{Pk}_j)$ and $\zeta \leftarrow \text{Res}^V(M, \text{Sk}_j, \theta, \text{Pk}_i, \text{Pk}_j, \text{Apk})$.*

On the Resolution Protocol: To resolve a partial signature σ , V sends it to the arbitrator, which runs Res^A to convert it into an intermediate value θ and returns to V . The verifier then runs Res^V to recover the full signature ζ from θ . In this way the arbitrator does not learn the final output of the resolution. Furthermore, as in the definition of *perfect ambiguity* (Def. 3), we require that the arbitrator does not know whether the submitted partial signature contains a valid full signature on M of the signer. In Sec. 6 we explain in more details how the resolution of our proposed P^2OFE protocol works in practice.

Remark. We stress that in P²OFE, giving a message/contract M itself to the arbitrator in clear does not harm the signer, since guaranteed by the perfect ambiguity, the arbitrator cannot confirm or convince others that the signer has *signed* M .

3.2 Security Models

We now study the security properties that a P²OFE protocol should satisfy. First of all, the correctness of P²OFE can be naturally defined, and we omit it here. A secure P²OFE protocol should satisfy the following properties: *resolution ambiguity*, *signer ambiguity*, *perfect ambiguity*, *security against signers* and *security against the arbitrator*. Below we introduce them individually, where for simplicity we omit the generation of system parameters PM. All the security properties of P²OFE are defined in the *certified-key* model [5, 18], in which an adversary can query an oracle O_{KR} which takes as input a key pair (Pk, Sk) and outputs 1 if it is in the range of algorithm U_{kg} and 0 otherwise. For simplicity we omit O_{KR} in the following experiments.

Resolution Ambiguity: The property states that full signatures output by the signer should be computationally indistinguishable from those output by the verifier at the end of the resolution protocol. Let

$$\begin{aligned}\Delta_0 &\stackrel{\text{def}}{=} \{\zeta \leftarrow \text{Sig}(M, Sk_i, Pk_i, Pk_j, Apk)\}, \text{ and} \\ \Delta_1 &\stackrel{\text{def}}{=} \{\zeta \leftarrow \text{Res}^V(M, Sk_j, \theta, Pk_i, Pk_j, Apk)\},\end{aligned}$$

where $\theta \leftarrow \text{Res}^A(\text{Ask}, \sigma, Pk_i, Pk_j)$ and $\sigma \leftarrow \text{PSig}(M, Sk_i, Pk_i, Pk_j, Apk)$. A protocol is resolution ambiguous if Δ_0 and Δ_1 are computationally indistinguishable.

Signer Ambiguity: Before giving the definition of signer ambiguity, we describe a new p.p.t. algorithm FPSig that is run by the verifier to simulate the signer's partial signature. The algorithm is similar with PSig . It takes as input $(M, Sk_j, Pk_i, Pk_j, Apk)$ and outputs a partial signature valid under Pk_i, Pk_j and Apk . We require that there exists an algorithm FPSig such that for any p.p.t. adversary \mathcal{A} , which models a dishonest signer, succeeds with at most negligible advantage in the following experiment Exp_{sa} :

$$\begin{aligned}(Apk, Ask) &\leftarrow \text{Akg}(PM) \\ (Pk_\gamma, Sk_\gamma) &\leftarrow \text{Ukg}(PM, Apk), \forall \gamma \in \{0, 1\}\end{aligned}$$

$$\begin{aligned}
(M^*, \mathcal{Y}) &\leftarrow \mathcal{A}^{O_{\text{Res}^A}}(\{(\text{Pk}_\gamma, \text{Sk}_\gamma)\}_{\gamma=0}^1, \text{Apk}) \\
b &\leftarrow \{0, 1\} \\
\sigma^* &\leftarrow \begin{cases} \text{PSig}(M^*, \text{Sk}_0, \text{Pk}_0, \text{Pk}_1, \text{Apk}) & \text{if } b = 0 \\ \text{FPSig}(M^*, \text{Sk}_1, \text{Pk}_0, \text{Pk}_1, \text{Apk}) & \text{if } b = 1 \end{cases} \\
b' &\leftarrow \mathcal{A}^{O_{\text{Res}^A}}(\mathcal{Y}, \sigma^*) \\
\text{Succ. of } \mathcal{A} &:= [b' = b \wedge (M^*, \sigma^*, \text{Pk}_0, \text{Pk}_1) \notin \mathcal{Q}(\mathcal{A}, O_{\text{Res}^A}) \\
&\quad \wedge (M^*, \sigma^*, \text{Pk}_1, \text{Pk}_0) \notin \mathcal{Q}(\mathcal{A}, O_{\text{Res}^A})],
\end{aligned}$$

where

- O_{Res^A} takes as input $(M, \sigma, \text{Pk}_i, \text{Pk}_j)$ and outputs the corresponding intermediate signature θ or \perp indicating the failure of conversion; and
- $\mathcal{Q}(\mathcal{A}, O_{\text{Res}^A})$ is the set of queries that \mathcal{A} submitted to oracle O_{Res^A} .

The advantage of \mathcal{A} in the experiment is defined as $\text{Adv}_{\text{sa}}^A(1^k) := |\Pr[\text{Succ}_{\text{sa}}] - 1/2|$, where Succ_{sa} denotes the event that \mathcal{A} succeeds in the experiment \mathbf{Exp}_{sa} .

Definition 2 (Signer Ambiguity). *A P²OFE protocol is signer ambiguous if there is no p.p.t. \mathcal{A} , such that $\text{Adv}_{\text{sa}}^A(1^k)$ is non-negligible in the security parameter k .*

Perfect Ambiguity: It basically says that given a partial signature, even the arbitrator cannot assert which users are involved in the signature exchange. Technically, we require that the distinguisher (which could be the arbitrator) is unable to tell whether the given signature was generated honestly by signer A w.r.t. the verifier B , or randomly selected from the signature space. We need a p.p.t. algorithm Sim that is run by the public to simulate signatures of A and B . The algorithm takes as input $(\text{Apk}, \text{Pk}_i, \text{Pk}_j)$ and outputs a simulated partial signature of the signer U_i w.r.t. the verifier U_j . Formally, we require that there exists an algorithm Sim such that for any p.p.t. adversary \mathcal{A} , it succeeds in the following experiment \mathbf{Exp}_{pa} with only negligible advantage:

$$\begin{aligned}
(\text{Apk}, \text{Ask}) &\leftarrow \text{Akg}(\text{PM}) \\
(\text{Pk}_\gamma, \text{Sk}_\gamma) &\leftarrow \text{Ukg}(\text{PM}, \text{Apk}), \quad \forall \gamma \in \{0, 1\} \\
(M^*, \mathcal{Y}) &\leftarrow \mathcal{A}^{O_{\text{PSig}^V}, O_{\text{FPSig}}, O_{\text{Res}^V}}(\text{Pk}_0, \text{Sk}_0, \text{Pk}_1, \text{Apk}, \text{Ask}) \\
b &\leftarrow \{0, 1\} \\
\sigma^* &\leftarrow \begin{cases} \text{PSig}(M^*, \text{Sk}_0, \text{Pk}_0, \text{Pk}_1, \text{Apk}), & \text{if } b = 0 \\ \text{Sim}(\text{Apk}, \text{Pk}_0, \text{Pk}_1) & , \text{ if } b = 1 \end{cases}
\end{aligned}$$

$$\begin{aligned}
b' &\leftarrow \mathcal{A}^{O_{\text{PSig}^V}, O_{\text{FPSig}}, O_{\text{Res}^V}}(\mathcal{I}, \sigma^*) \\
\theta^* &\leftarrow \text{Res}^A(M^*, \text{Ask}, \sigma^*, \text{Pk}_0, \text{Pk}_1) \\
\text{Succ. of } \mathcal{A} &:= [b' = b \wedge (M^*, \theta^*, \text{Pk}_0) \notin \mathcal{Q}(\mathcal{A}, O_{\text{Res}^V})],
\end{aligned}$$

where

- O_{PSig^V} takes as input (M, Pk') , and outputs a partial signature $\sigma \leftarrow \text{PSig}(M, \text{Sk}_1, \text{Pk}_1, \text{Pk}', \text{Apk})$;
- O_{FPSig} takes as input (M, Pk') , and outputs a simulated partial signature, e.g. $\sigma \leftarrow \text{FPSig}(M, \text{Sk}_1, \text{Pk}', \text{Pk}_1, \text{Apk})$;
- O_{Res^V} takes as input (M, θ, Pk') , and outputs the full signature $\zeta \leftarrow \text{Res}^V(M, \text{Sk}_1, \theta, \text{Pk}', \text{Pk}_1, \text{Apk})$; and
- $\mathcal{Q}(\mathcal{A}, O_{\text{Res}^V})$ is the set of queries that \mathcal{A} submitted to oracle O_{Res^V} .

The advantage of \mathcal{A} in the experiment is defined as $\text{Adv}_{\text{pa}}^A(1^k) := |\Pr[\text{Succ}_{\text{pa}}] - 1/2|$, where Succ_{pa} denotes the event that \mathcal{A} succeeds in the experiment \mathbf{Exp}_{pa} .

Definition 3 (Perfect Ambiguity). *A P²OFE protocol is perfect ambiguous if there is no p.p.t. adversary \mathcal{A} such that $\text{Adv}_{\text{pa}}^A(1^k)$ is non-negligible in the security parameter k .*

Remark. Notice that in the experiment above we do not give the adversary access to an oracle which returns full signatures of the verifier (with public key Pk_1). The oracle can be implemented by composing O_{PSig^V} and O_{Res^V} as well as the knowledge of Ask .

The simulation algorithm Sim does not take any secret input and can be run by anyone to simulate a partial signature that looks indistinguishable from a real one. Guaranteed by the perfect ambiguity, without the knowledge of the intended verifier's secret key, no one is able to determine whether a given partial signature does come from the signer. Due to the public simulatability, even the arbitrator cannot assert and convince others that the signer indeed signed the message M . In other words, the signer could deny the generation of a partial signature.

Security against Signers: To protect the verifier from being cheated, the signer should be unable to produce a partial signature such that it can pass the partial verification, but the resolution fails to output a valid full signature. Formally, we consider the following experiment $\mathbf{Exp}_{\text{sas}}$:

$$\begin{aligned}
(\text{Apk}, \text{Ask}) &\leftarrow \text{Akg}(\text{PM}) \\
(\text{Pk}_1, \text{Sk}_1) &\leftarrow \text{Ukg}(\text{PM}, \text{Apk})
\end{aligned}$$

$$\begin{aligned}
(M^*, \text{Pk}_0, \sigma^*) &\leftarrow \mathcal{A}^{O_{\text{FPSig}}, O_{\text{Res}}}(\text{Pk}_1, \text{Apk}) \\
\theta^* &\leftarrow \text{Res}^A(M^*, \text{Ask}, \sigma^*, \text{Pk}_0, \text{Pk}_1) \\
\zeta^* &\leftarrow \text{Res}^V(M^*, \text{Sk}_1, \theta^*, \text{Pk}_0, \text{Pk}_1, \text{Apk}) \\
\text{Succ. of } \mathcal{A} &:= [\text{PVer}(M^*, \sigma^*, \text{Pk}_0, \text{Pk}_1, \text{Apk}, \text{Sk}_1) = 1 \\
&\quad \wedge \text{Ver}(M^*, \zeta^*, \text{Pk}_0, \text{Pk}_1, \text{Apk}) = 0 \\
&\quad \wedge (M^*, \text{Pk}_0) \notin \mathcal{Q}(\mathcal{A}, O_{\text{FPSig}})],
\end{aligned}$$

where

- $O_{\text{Res}} = \langle O_{\text{Res}^A}, O_{\text{Res}^V} \rangle$ takes as (M, σ, Pk') , and outputs the corresponding full signature ζ (that is valid w.r.t. the signer's public key Pk' , the verifier's public key Pk_1 and Apk) or \perp ; and
- $\mathcal{Q}(\mathcal{A}, O_{\text{FPSig}})$ is the set of queries that \mathcal{A} submitted to the oracle O_{FPSig} .

The advantage of \mathcal{A} in the experiment is defined as $\text{Adv}_{\text{sas}}^{\mathcal{A}}(1^k) := \Pr[\text{Succ}_{\text{sas}}]$, where Succ_{sas} denotes the event that \mathcal{A} succeeds in the experiment $\mathbf{Exp}_{\text{sas}}$.

Definition 4 (Security against Signers). *A P²OFE protocol is secure against signers if there is no p.p.t. adversary \mathcal{A} such that $\text{Adv}_{\text{pa}}^{\mathcal{A}}(1^k)$ is non-negligible in the security parameter k .*

Security against the Arbitrator: To be fair for the signer, no one but the signer, should be able to produce valid signatures on behalf of the signer. Formally, we consider the following experiment $\mathbf{Exp}_{\text{saa}}$:

$$\begin{aligned}
(\text{Apk}, \text{Ask}) &\leftarrow \text{Akg}(\text{PM}) \\
(\text{Pk}_0, \text{Sk}_0) &\leftarrow \text{Ukg}(\text{PM}, \text{Apk}) \\
(M^*, \text{Pk}_1, \zeta^*) &\leftarrow \mathcal{A}^{O_{\text{PSig}}}(\text{Pk}_0, \text{Apk}, \text{Ask}) \\
\text{Succ. of } \mathcal{A} &:= [\text{Ver}(M^*, \zeta^*, \text{Pk}_0, \text{Pk}_1, \text{Apk}) = 1 \wedge (M^*, \text{Pk}_1) \notin \mathcal{Q}(\mathcal{A}, O_{\text{PSig}})],
\end{aligned}$$

where

- O_{PSig} takes as input a message M and a public key Pk' and outputs $\sigma \leftarrow \text{PSig}(M, \text{Sk}_0, \text{Pk}_0, \text{Pk}', \text{Apk})$; and
- $\mathcal{Q}(\mathcal{A}, O_{\text{PSig}})$ is the set of queries that \mathcal{A} submitted to O_{PSig} .

The advantage of \mathcal{A} in the experiment is defined as $\text{Adv}_{\text{saa}}^{\mathcal{A}}(1^k) := \Pr[\text{Succ}_{\text{saa}}]$, where Succ_{saa} denotes the event that \mathcal{A} succeeds in $\mathbf{Exp}_{\text{saa}}$.

Definition 5 (Security against the Arbitrator). *A P²OFE protocol is secure against the arbitrator if there is no p.p.t. adversary \mathcal{A} such that $\text{Adv}_{\text{saa}}^{\mathcal{A}}(1^k)$ is non-negligible in the security parameter k .*

Remark 1. Security against the arbitrator assumes the adversary (including the arbitrator) is malicious and is allowed to try all kinds of ways to forge the signer’s signature. This is for protecting the signer to the maximum extent. However, the arbitrator is still assumed to function normally as prescribed in practice, i.e. to honestly resolve signatures according to the users’ needs.

3.3 Differences from Other Variants of OFE

In this part we summarize the differences between P²OFE and (other variants of) OFE. Table 1 shows the comparison. In the table, “Ambiguity of σ Before Resolution” (resp. “Ambiguity of σ After Resolution”) refers to that given only a partial signature σ , whether anyone (including the arbitrator) could convince others before (resp. after) the resolution takes place that the signer has signed the message. We denote by “ $\sqrt{2}$ ” that σ is ambiguous in the sense that either the signer or the verifier could generate σ , and by “ $\sqrt{\infty}$ ” that σ is ambiguous in the sense that everyone could be the source of σ .

Table 1. Comparison with other variants of OFE

Variants	Ambiguity of σ Before Resolution	Ambiguity of σ After Resolution
OFE	\times	\times
AOFE	$\sqrt{2}$	\times
PAOFE	$\sqrt{\infty}$	\times
P ² OFE	$\sqrt{\infty}$	$\sqrt{\infty}$

The partial signature in traditional OFE [3, 10, 20] is publicly verifiable, and everyone is able to tell from it the fact that the signer signed the message. In the enhanced variant AOFE [15, 16, 18, 19], although the partial signature is ambiguous, however, anyone is still able to confirm that the given partial signature was generated by either the signer or the verifier. In PAOFE [30], the ambiguity is further improved. No one but the verifier is able to tell from the given partial signature who the real signer is. However, no matter the partial signature is ambiguous or not, the arbitrator in these variants has a full copy of the signer’s full signature after the resolution.

In our new notion of OFE, given only the partial signature, neither the arbitrator nor the verifier is able to find out by itself who the signer is. Thus, the arbitrator could not convince others that the signer did sign the

message. Furthermore, this also holds even after the resolution in P²OFE, as guaranteed by the perfect ambiguity of P²OFE.

4 Mathematical Assumptions

The P²OFE protocol is bilinear pairing based, and its security is based on the Decision Linear and Strong Diffie-Hellman assumptions, which are reviewed as follows:

Bilinear Pairing. Let \mathbb{G}, \mathbb{G}_T be two cyclic groups of prime order p , and g be a random generator of \mathbb{G} . The map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear pairing if (1) Bilinear: $\forall u, v \in \mathbb{Z}_p, \hat{e}(g^u, g^v) = \hat{e}(g, g)^{uv}$; (2) Non-degenerate: $\hat{e}(g, g) \neq 1_T$, where 1_T is the identity element of group \mathbb{G}_T ; and (3) Computable: there exists a polynomial-time algorithm for computing $\hat{e}(U, V)$ for any $U, V \in \mathbb{G}$.

Definition 6 (Decision Linear Assumption [8]). Let \mathbb{G}, \mathbb{G}_T be cyclic groups of prime order p , and g be a random generator of \mathbb{G} . Let $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear pairing. The Decision Linear (DLIN) assumption in the context of $(\mathbb{G}, \mathbb{G}_T, \hat{e}, p, g)$ says that there is no p.p.t. algorithm \mathcal{A} such that for all $F, G \leftarrow \mathbb{G}, s, t, z \leftarrow \mathbb{Z}_p$,

$$\left| \Pr[\mathcal{A}(F, G, F^s, G^t, g^{s+t}) = 1] - \Pr[\mathcal{A}(F, G, F^s, G^t, g^z) = 1] \right| \leq \text{negl}(k),$$

where $\text{negl}(\cdot)$ is a negligible function in the security parameter k , and the probabilities are taken over the choices of $F, G \in \mathbb{G}, s, t, z \in \mathbb{Z}_p$ and the random bits consumed by \mathcal{A} .

Definition 7 (Strong Diffie-Hellman Assumption [7]). Let \mathbb{G} be a cyclic group of prime order p , and g be a random generator of it. The ℓ -Strong Diffie-Hellman (ℓ -SDH) assumption says that there is no p.p.t. algorithm \mathcal{A} such that for all $x \leftarrow \mathbb{Z}_p$,

$$\Pr\left[Z = g^{\frac{1}{x+c}} \mid (Z, c) \leftarrow \mathcal{A}(g, g^x, g^{x^2}, \dots, g^{x^\ell})\right] \leq \text{negl}(k),$$

where $c \in \mathbb{Z}_p$, and the probability is taken over the choice of $x \in \mathbb{Z}_p$ and the random bits consumed by \mathcal{A} .

5 Our Protocol

In this section we present a concrete construction of P²OFE. Before presenting the concrete protocol, we give a high level description of how our protocol works.

5.1 High Level Idea

Briefly speaking, our protocol makes use of Boneh-Boyen short signature scheme (BB signature, for short) [7] and the tag-based public key encryption scheme [24]. It essentially follows the sign-then-encrypt paradigm. To generate a full signature ζ on message M , the signer simply runs the corresponding algorithm of BB signature scheme. To partially sign M , the signer first generates a BB signature $\zeta = (S, r)$ and encrypts ζ w.r.t. the arbitrator's public key using the tag-based encryption scheme while keeping r public. Let the ciphertext be e . Then the signer encrypts (part of) e under the intended verifier's public key again and obtains a new ciphertext c . The two encryptions are twisted together so that the arbitrator and the intended verifier can perform their own decryption, but cannot recover the signer's full signature alone. To prevent the adversary from making use of the resolution oracle to break the security of the protocol, we use a strong one-time signature scheme to sign the whole ciphertext and use the fresh one-time verification key as the tag in the tag-based encryption. To convince the verifier the validity of σ , the signer needs carry out a proof with the verifier.

In order to resolve a partial signature σ to a full one, the verifier sends σ to the arbitrator. The latter uses its secret key to do the first level decryption and returns the resulting value, which is a ciphertext of ζ . The verifier then extracts the full signature by performing another decryption using its own secret key.

5.2 The Protocol

Let \mathbb{G}, \mathbb{G}_T be two cyclic multiplicative groups of prime order p , g a random generator of \mathbb{G} , and $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear pairing. Let OTS be a strong one-time signature scheme and \mathcal{VK} be the space of one-time verification keys. Let $H : \mathbb{G}^5 \times \mathcal{VK} \rightarrow \mathbb{Z}_p$ be a collision-resistant hash function. Our P²OFE protocol works as follows. In the protocol we assume the message space is \mathbb{Z}_p , which can be easily extended to $\{0, 1\}^*$ by applying a collision-resistant hash function onto the message.

Akg. The arbitrator chooses at random $\xi_1, \xi_2 \leftarrow \mathbb{Z}_p$, $K, L \in \mathbb{G}$, and computes $F = g^{1/\xi_1}$, $G = g^{1/\xi_2}$. It sets $\mathbf{Apk} = (F, G, K, L)$ and $\mathbf{Ask} = (\xi_1, \xi_2)$.

Ukg. The user U_i chooses at random $x_i, y_i, \xi_{i1}, \xi_{i2} \in \mathbb{Z}_p$ and computes $X_i = g^{x_i}$, $Y_i = g^{y_i}$, $F_i = g^{1/\xi_{i1}}$ and $G_i = g^{1/\xi_{i2}}$. The user sets $\mathbf{Pk}_i = (X_i, Y_i, F_i, G_i, K_i, L_i)$ and $\mathbf{Sk}_i = (x_i, y_i, \xi_{i1}, \xi_{i2})$.

PSig. Given a message M , the signer U_i generates its partial signature for the verifier U_j as follows.

1. Select at random $r, s, t, s', t' \in \mathbb{Z}_p$.
2. Run $\text{OTS.Kg}(1^k)$ to generate a one-time key pair $(\text{otvk}, \text{otsk})$.
3. Compute

$$\begin{aligned} c_1 &= F_j^{s'}, & c_2 &= G_j^{t'}, & S &= g^{1/(x_i+M+y_i \cdot r)}, \\ e_1 &= F^s, & e_2 &= G^t, & e_3 &= Sg^{s+t}g^{s'+t'}, & \alpha &= \text{H}(c_1, c_2, e_1, e_2, e_3, \text{otvk}), \\ c_4 &= (g^\alpha K_j)^{s'}, & c_5 &= (g^\alpha L_j)^{t'}, & e_4 &= (g^\alpha K)^s, & e_5 &= (g^\alpha L)^t, \\ \delta &= \text{OTS.Sig}(\text{otsk}, M \parallel \text{Pk}_i \parallel \text{Pk}_j \parallel \mathbf{c} \parallel \mathbf{e} \parallel r), \end{aligned}$$

where $\mathbf{e} = (e_1, e_2, e_3, e_4, e_5)$ and $\mathbf{c} = (c_1, c_2, c_4, c_5)$.

If $x_i + M + y_i r = 0 \pmod p$, the signer chooses another r and repeats the process. Its partial signature on M is $\sigma = (\mathbf{c}, \mathbf{e}, r, \text{otvk}, \delta)$.

PVer. Given a partial signature $\sigma = (\mathbf{c}, \mathbf{e}, r, \text{otvk}, \delta)$, U_i and U_j check the well-formedness of the signature locally, and do nothing if either of the following does not hold:

$$\hat{e}(e_4, F) = \hat{e}(e_1, g^\alpha K), \quad (1)$$

$$\hat{e}(e_5, G) = \hat{e}(e_2, g^\alpha L), \quad (2)$$

$$\hat{e}(c_4, F_j) = \hat{e}(c_1, g^\alpha K_j), \quad (3)$$

$$\hat{e}(c_5, G_j) = \hat{e}(c_2, g^\alpha L_j), \quad (4)$$

$$\text{OTS.Sig}(M \parallel \text{Pk}_i \parallel \text{Pk}_j \parallel \mathbf{c} \parallel \mathbf{e} \parallel r, \text{otvk}, \delta) = 1, \quad (5)$$

where $\alpha = \text{H}(c_1, c_2, e_1, e_2, e_3, \text{otvk})$. Then they carry out the following witness-indistinguishable proof to show that σ contains a valid BB signature of either U_i or U_j :

$$\begin{aligned} \Pi \stackrel{\text{def}}{=} PK \left\{ (s, t, s', t') : c_1 &= F_j^{s'} \wedge c_2 = G_j^{t'} \wedge e_1 = F^s \wedge e_2 = G^t \right. \\ &\wedge (\hat{e}(e_3 \cdot g^{-s-t-s'-t'}, X_i g^M Y_i^r) = \hat{e}(g, g)) \\ &\left. \vee \hat{e}(e_3 \cdot g^{-s-t-s'-t'}, X_j g^M Y_j^r) = \hat{e}(g, g) \right\}. \quad (6) \end{aligned}$$

Sig. To generate a full signature on message M for the verifier U_j , the signer U_i randomly selects $r \in \mathbb{Z}_p$, and computes

$$S = g^{1/(x_i+M+y_i \cdot r)}.$$

Again, in case that $x_i + M + y_i r = 0 \pmod p$, it chooses another r and repeats the computation. Its full signature on M is $\zeta = (S, r)$.

Ver. Given (M, ζ) where $\zeta = (S, r)$, the verifier checks if

$$\hat{e}(S, X_i g^M Y_i^r) = \hat{e}(g, g). \quad (7)$$

It outputs 1 if the equation holds, and 0 otherwise.

Res^A. Given $(M, \sigma, \text{Pk}_i, \text{Pk}_j)$ where $\sigma = (c, e, r, \text{otvk}, \delta)$, the arbitrator returns \perp if either Eq. (1), (2), (3), (4) or (5) fails to hold; otherwise, it computes

$$c_3 = e_3 e_1^{-\xi_1} e_2^{-\xi_2}, \quad (8)$$

and returns $\theta = (c_1, c_2, c_3, c_4, c_5, e_3, r, \text{otvk})$.

Res^V. Given $(M, \theta, \text{Pk}_i, \text{Pk}_j)$, where $\theta = (c_1, c_2, c_3, c_4, c_5, e_3, r, \text{otvk})$, the verifier outputs \perp if either Eq. (3) or (4) fails to hold; otherwise, it computes

$$S = c_3 c_1^{-\xi_{j1}} c_2^{-\xi_{j2}}. \quad (9)$$

It outputs $\zeta = (S, r)$ if Eq. (7) holds, and \perp otherwise.

5.3 Security

Below we show the security of our P²OFE protocol based on the assumptions described in Sec. 4.

Theorem 1. *Our P²OFE protocol is resolution ambiguous.*

Proof. Notice that the full signature output by the signer and that output by the resolution protocol are of the form $\zeta = (S, r)$, which is a Boneh-Boyen signature on the message M . Therefore, our P²OFE protocol is perfectly resolution ambiguous. \square

Theorem 2. *Our P²OFE protocol is signer ambiguous if DLIN assumption holds, H is collision-resistant and OTS is a strong one-time signature scheme.*

Theorem 3. *Our P²OFE protocol is perfect ambiguous, if DLIN assumption holds, H is collision resistant and OTS is a strong one-time signature scheme.*

Theorem 4. *Our P²OFE protocol is secure against signers, if SDH assumption holds, and Π is sound and witness-indistinguishable.*

Theorem 5. *Our P²OFE protocol is secure against the arbitrator if SDH assumption holds.*

Due to the page limit we defer the proofs to the full version.

6 Resolution in Practice

In this section we describe one of the ways on how P²OFE runs in practice. Suppose the electronic contract that Alice and Bob want to secretly sign is M , and their semi-trusted third party is Ted. Recall that the contract M itself does not need to be secret, as anyone can prepare such a contract. Instead, signatures of Alice and Bob should be kept secret from others. Without their signatures, no one can confirm whether they have signed the contract or have really performed such a business deal.

Following the framework of optimistic fair exchange, Alice and Bob exchange their signatures on M . If everything goes well, they will receive the counterpart's full signature. Due to that a party might refuse to continue the run of the exchange protocol, or that the internet connection might become down, there are two cases in which a dispute will occur between the two parties, as below:

1. after sending out the first-move message, which is Alice's partial signature σ_A on M , Alice receives nothing;
2. after sending out the second-move message, which is Bob's full signature ζ_B on M , Bob receives nothing.

Let us focus on the latter case first. In this case, Bob can resort to the arbitrator, Ted, for converting σ_A to the full signature of Alice. Before the conversion, Bob has to show the fulfillment of his obligation. Traditionally, this can be done by sending his full signature ζ_B to the arbitrator, the validity of which can be verified publicly. However, this will let the arbitrator confirm, and even show to others the involvement of Bob as ζ_B shows that Bob indeed signed M . This is undesired in some sensitive applications. To avoid this problem, Bob instead sends his partial signature σ_B on M to Ted, and carries out a *zero-knowledge* (or *designated-verifier* [23]) proof of knowledge to convince Ted that σ_B does encapsulate his full signature on M . If he accepts the proof, Ted runs Res^A on input σ_A (as well as M) to obtain the intermediate value θ_A and sends it to Bob. In the meanwhile, he also runs Res^A on input σ_B to obtain Bob's intermediate value θ_B and sends it to Alice, in order to avoid the case in which Bob tried to cheat at the end of the first move and did not ever send his full signature to Alice. Figure 3 shows how the resolution of P²OFE works in practice, where Π_B is the proof run by B to show the fulfillment of his obligation.

Now let us go back to the former case. If Bob does not try to cheat and simply aborts the protocol, guaranteed by the signer ambiguity, Bob

does not learn anything from Alice partial signature, as long as Ted does not collude with Bob. In this case, neither Alice nor Bob obtains their counterpart's (full) signature. However, if Bob tries to cheat and asks Ted for the resolution, according to the aforementioned resolution procedure, Bob still needs to provide his partial signature and a proof to support the validity of his signature.

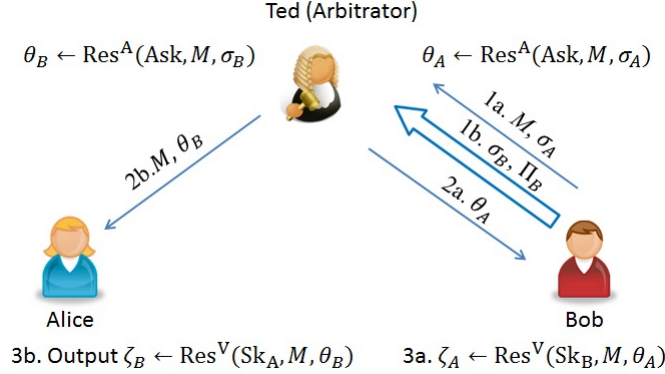


Fig. 3. P²OFE: Resolution in Practice

It should be noticed that the message signed by Alice and that signed by Bob are not required to be the same, depending on the applications. In applications where they need sign different messages, it suffices that Alice (resp. Bob) runs algorithms P^{Sig}, Sig on input M_A (resp. M_B) and runs P^{Ver}, Ver, Res^V on input M_B (resp. M_A).

7 Conclusion

We introduced the notion P²OFE for achieving the privacy preserving property not just against a semi-trusted honest-but-curious arbitrator, but also against a completely malicious arbitrator. This is the first time in the context of OFE that signer privacy can be ensured even *after* the resolution. We also proposed an efficient concrete construction of P²OFE with each of its full signatures being as simple as a Boneh-Boyen short signature. Based on SDH and DLIN assumptions, we also showed its security under the security model we defined without random oracles. As of practical interest, we further demonstrated how the resolution can actually work in practice.

Acknowledgements

We'd like to thank the anonymous reviewers for their invaluable comments. This work is supported by the National Natural Science Foundation of China (No. 61103232), the Guangdong Natural Science Foundation (No. S2013010011859), the Research Fund for the Doctoral Program of Higher Education of China (No. 20114404120027), and the Foundation for Distinguished Young Talents in Higher Education of Guangdong, China (No. LYM11033). D. S. Wong is supported by a grant from the RGC of the HKSAR, China (Project No. CityU 121512). W. Susilo is supported by ARC Future Fellowship FT0991397.

References

1. N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *ACM Conference on Computer and Communications Security*, pages 7–17. ACM, 1997.
2. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures (extended abstract). In *EUROCRYPT98*, volume 1403 of *LNCS*, pages 591–606. Springer, 1998.
3. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communication*, 18(4):593–610, 2000.
4. G. Avoine and S. Vaudenay. Optimistic fair exchange based on publicly verifiable secret sharing. In *ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*, pages 74–85. Springer, 2004.
5. B. Barak, R. Canetti, J. B. Nielsen, and R. Pass. Universally composable protocols with relaxed set-up assumptions. In *FOCS04*, pages 186–195. IEEE Computer Society, 2004.
6. M. Belenkiy, M. Chase, C. C. Erway, J. Jannotti, A. Küpçü, A. Lysyanskaya, and E. Rachlin. Making p2p accountable without losing privacy. In *WPES*, pages 31–40. ACM, 2007.
7. D. Boneh and X. Boyen. Short signatures without random oracles. In *EUROCRYPT04*, volume 3027 of *LNCS*, pages 56–73. Springer, 2004.
8. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.
9. D. Chaum. Designated confirmer signatures. In *EUROCRYPT94*, volume 950 of *LNCS*, pages 86–91. Springer, 1995.
10. Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic fair exchange in a multi-user setting. In *PKC07*, volume 4450 of *LNCS*, pages 118–133. Springer, 2007.
11. Y. Dodis and L. Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In *ACM Workshop on Digital Rights Management, DRM 2003*, pages 47–54. ACM, 2003.
12. Y. Dodis and D. H. Yum. Time capsule signatures. In *FC05*, volume 3570 of *LNCS*, pages 57–71. Springer, 2005.
13. J. A. Garay, M. Jakobsson, and P. MacKenzie. Abuse-free optimistic contract signing. In *CRYPTO99*, volume 1666 of *LNCS*, pages 449–466. Springer, 1999.

14. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT08*, volume 4965 of *LNCS*, pages 415–432. Springer, 2008.
15. Q. Huang, D. S. Wong, and W. Susilo. A new construction of designated confirmer signature and its application to optimistic fair exchange - (extended abstract). In *Pairing10*, volume 6487 of *LNCS*, pages 41–61. Springer, 2010.
16. Q. Huang, D. S. Wong, and W. Susilo. Efficient designated confirmer signature and DCS-based ambiguous optimistic fair exchange. *IEEE Transactions on Information Forensics and Security*, 6(4):1233–1247, 2011.
17. Q. Huang, D. S. Wong, and W. Susilo. Group-oriented fair exchange of signatures. *Information Sciences*, 181(16):3267–3283, 2011.
18. Q. Huang, D. S. Wong, and W. Susilo. The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles. In *PKC12*, volume 7293 of *LNCS*, pages 120–137. Springer, 2012.
19. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Ambiguous optimistic fair exchange. In *ASIACRYPT08*, volume 5350 of *LNCS*, pages 74–89. Springer, 2008.
20. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In *CT-RSA08*, volume 4964 of *LNCS*, pages 106–120. Springer, 2008.
21. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. A new efficient optimistic fair exchange protocol without random oracles. *International Journal of Information Security*, 11(1):53–63, 2012.
22. X. Huang, Y. Mu, W. Susilo, W. Wu, J. Zhou, and R. H. Deng. Preserving transparency and accountability in optimistic fair exchange of digital signatures. *IEEE Transactions on Information Forensics and Security*, 6(2):498–512, 2011.
23. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *EUROCRYPT96*, volume 1070 of *LNCS*, pages 143 – 154. Springer, 1996.
24. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC06*, volume 3876 of *LNCS*, pages 581–600. Springer, 2006.
25. A. K upc u and A. Lysyanskaya. Optimistic fair exchange with multiple arbiters. In *ESORICS 2010*, volume 6345 of *Lecture Notes in Computer Science*, pages 488–507. Springer, 2010.
26. A. K upc u and A. Lysyanskaya. Usable optimistic fair exchange. *Computer Networks*, 56(1):50–63, 2012.
27. A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In *EUROCRYPT04*, volume 3027 of *LNCS*, pages 74–90. Springer, 2004.
28. J. M. Park, E. K. Chong, and H. J. Siegel. Constructing fair-exchange protocols for e-commerce via distributed computation of RSA signatures. In *PODC 2003*, pages 172–181. ACM, 2003.
29. G. Wang. An abuse-free fair contract signing protocol based on the RSA signature. *IEEE Transactions on Information Forensics and Security*, 5(1):158–168, March 2010.
30. Y. Wang, M. H. Au, and W. Susilo. Perfect ambiguous optimistic fair exchange. In *ICICS12*, volume 7618 of *LNCS*, pages 142–153. Springer, 2012.