1-2012

A new efficient optimistic fair exchange protocol without random oracles

Qiong HUANG

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Duncan S. WONG

Willy SUSILO

Citation

HUANG, Qiong; YANG, Guomin; WONG, Duncan S.; and SUSILO, Willy. A new efficient optimistic fair exchange protocol without random oracles. (2012). *International Journal of Information Security*. 11, (1), 53-63.

Available at: https://ink.library.smu.edu.sg/sis_research/7351

Int. J. Inf. Secur. (2012) 11:53–63
DOI 10.1007/s10207-011-0152-3

REGULAR CONTRIBUTION

A new efficient optimistic fair exchange protocol without random oracles

Qiong Huang · Guomin Yang · Duncan S. Wong · Willy Susilo

Published online: 1 December 2011

Abstract Optimistic fair exchange (OFE) is a kind of protocols to solve the problem of fair exchange between two parties. Most of the previous work on this topic are provably secure in the random oracle model. In this work, we propose a new construction of OFE from another cryptographic primitive, called *time capsule signature*. The construction is efficient and brings almost no overhead other than the primitive itself. The security of our new construction is based on that of the underlying primitive without relying on the random oracle heuristic. Applying our generic construction to the time capsule signature scheme recently proposed by Libert and Quisquater, we obtain a new concrete and efficient OFE construction secure based on Computational Diffie–Hellman assumption in the standard model.

Qiong Huang was supported by the National Natural Science Foundation of China under Grant 61103232; Duncan S. Wong was supported by grants from CityU (Project No. 7002585, 7002711); and Willy Susilo was supported by the Australian Research Council Future Fellowship FT0991397.

Q. Huang (✉)
College of Informatics, South China Agricultural University, Guangzhou, China
e-mail: csqhuang@alumni.cityu.edu.hk

G. Yang
Temasek Laboratories, National University of Singapore, Singapore, Singapore
e-mail: tslyg@nus.edu.sg

D. S. Wong
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong, China
e-mail: duncan@cityu.edu.hk

W. Susilo
School of Computer Science and Software Engineering, University of Wollongong, Northfields Avenue, Wollongong, Australia
e-mail: wsusilo@uow.edu.au

Keywords Optimistic fair exchange · Time capsule signature · Standard model · CDH assumption

1 Introduction

One of the major challenges in e-commerce is the establishment of pay-per-use applications for digital services, which are services that can be entirely rendered via an electronic network. Examples include the transfer of digital money, the delivery of video or audio data and provision of telephone and Internet access. A common characteristic of these types of services is that they normally *cannot be revoked*,i.e., once the service has been granted, then the service provider has *no* mechanism to force the recipient to return it. Therefore, the exchange of two digital services must occur *simultaneously* to ensure *fairness* for both parties. Unfortunately, real simultaneousness cannot be achieved in general since digital services cannot be granted simultaneously. This is due to the fact that any form of data transmitted requires some transmission time. The exchange of digital signatures constitutes an important part of any e-commerce transaction via the Internet, where participants do not trust each other. When the signatures are on a common piece of text, this is often referred to as a contract signing protocol.

Optimistic fair exchange (OFE), introduced by Asokan, Schunter and Waidner [1], is a kind of protocols to solve the problems in fairly exchanging items between two parties, say Alice and Bob. In such a protocol, there is an arbitrator who is *semi-trusted* by Alice and Bob and involves only if one party attempts to cheat the other or crashes. Let's consider the following scenario, in which Alice wants to purchase a software from Bob's online shop. Alice first partially authenticates a message saying that she allows Bob to obtain the money from her bank account. After checking the validity

of Alice's partial signature, Bob delivers the software to her. Later, if Alice is honest, she will send her full signature to Bob, with which Bob can collect the money from the bank. If Alice is dishonest and refuses to send back her full signature, Bob will turn to the arbitrator for help. He shows to the arbitrator the evidence of fulfilling his obligation, who will then resolve Alice's partial signature into a full one, and send it to Bob. With the full signature, Bob then can complete the transaction and obtain the money from Alice's bank account.

Since the introduction, OFE has attracted many researchers' attention, such as [2–4,11,14,15,17,22,23,27,28,31,33,34] and so on. Most of these works focus on single-user setting, in which there are only one signer and one verifier, as well as an arbitrator. This setting is stand-alone and does not reflect the real world very well. A more practical setting is the multi-user setting [14], in which there are multiple signers and verifiers, as well as an arbitrator. Each party could collude with other parties in order to cheat their counterparts. It is proved that security in single-user setting does not imply security in multi-user setting [14]. Until now there are only a couple of OFE schemes provably secure in multi-user setting, e.g., [14,22], and some of them are secure in the random oracle model [6] only. It is well known that security in this model is not preserved when random oracles are replaced with real-life hash functions [12].

1.1 Our contributions

In this work, we propose a new approach to constructing schemes of optimistic fair exchange of digital signatures. We show that optimistic fair exchange schemes in multi-user setting can be generically constructed from time capsule signatures, a cryptographic primitive introduced by Dodis and Yum in [16]. The resulting OFE scheme is as efficient as the underlying time capsule signature scheme. Combining recent work on time capsule signature in the standard model due to Libert and Quisquater [24], we then get an optimistic fair exchange scheme which is secure based on Computational Diffie–Hellman (CDH) assumption without random oracles in the multi-user setting.

Table 1 below shows a brief comparison of our scheme with some existing results on OFE in the multi-user setting, in terms of signature sizes, underlying assumptions and the need of random oracle model. From the comparison, we can see that our scheme is the only one secure in the standard model, i.e., without random oracles and without assuming a common reference string. The security of our scheme is based on a very standard number-theoretic assumption, e.g., Computational Diffie–Hellman assumption. In Sect. 7, we will give a more detailed comparison.

1.2 Related works

(*Optimistic Fair Exchange*). Since the introduction, OFE has attracted many researchers' attention, such as [2–4,11,14,15,17,22,23,27,28,31,33,34] and so on. To name a few, Asokan et al. studied the fair exchange of digital signatures in [2]. Park et al. proposed an OFE scheme following the sequential two-party multi-signature paradigm [28], which was later broken by Dodis et al. [15]. Dodis et al. also proposed a repaired scheme, in which each user registers a key with the arbitrator. Micali used a chosen-ciphertext secure public key encryption scheme with recoverable randomness to build another OFE scheme [27], which was cryptanalyzed by Bao et al. [4].

Since most of signature schemes in the literature are provably secure in the random oracle model [6], in which all parties have oracle access to a truly random function, most of schemes of OFE of signatures have provable security in this model as well. However, such a model is only heuristic.

Table 1 A brief comparison with some existing results in the multi-user setting

	[14]	[22] **Inst 1**	[22] **Inst 2**	[22] **Inst 3**	Ours
PSig	$3t + 2\mathbb{Z}_q + 1\mathbb{Z}_n^*$	$2\mathbb{G}$	$1\mathbb{G} + 1\mathbb{Z}_{n'}$	$2\mathbb{G} + 2\mathbb{Z}_p$	$3\mathbb{G}$
Sig	$2t + 1\mathbb{Z}_q + 1\mathbb{Z}_n^*$	$8\mathbb{G}$	$12\mathbb{G} + 5\mathbb{Z}_{n'}$	$5\mathbb{G} + 5\mathbb{Z}_p$	$6\mathbb{G}$
Assump	RSA + DL	CDH	SDH + SGD	Poly-SDH	CDH
Model	ROM	CRS	CRS	CRS	STD

In the concrete scheme proposed in [14], n is an RSA modulus, p is a prime larger than n, q is a prime such that $q|p-1$, and t is an integer such that $2^t < q$

Inst 1 in [22] instantiates the conventional signature with Waters signature [32] and the ring signature scheme with Schacham-Waters scheme [30]

Inst 2 in [22] instantiates the conventional signature with Boneh-Boyen signature [8] and the ring signature scheme with Chandran-Groth-Sahai scheme [13]

Inst 3 in [22] instantiates both the conventional signature and the ring signature with Boyen's mesh signature [10]

In both **Inst 1** and **Inst 2**, the group $\mathbb{G}$ is a bilinear group of composite order $n' = p'q'$, instead of prime order

In our scheme, group $\mathbb{G}$ is of prime order p and $\mathbb{G}_T$ is the target group of the bilinear pairing

CDH Computational Diffie–Hellman assumption, *DL* discrete logarithm assumption, *SDH* strong Diffie–Hellman assumption, *SGD* subgroup decision assumption, *Poly-SDH* poly strong Diffie–Hellman assumption *CRS* common reference string model, *ROM* random oracle model, *STD* standard model

Provable security of schemes in this model doesn't guarantee anything about the security when the random oracles are replaced with real-life hash functions [12].

Prior to Dodis et al.'s work [14], almost all previous works on OFE considered the single-user setting only, in which there are only one signer and one verifier (along with an arbitrator). They considered a more practical setting, called the *multi-user setting*, in which there are many signers and many verifiers (along with an arbitrator), so that a dishonest party can collude with some other parties in an attempt of cheating another party [14]. Although security of public key encryption and that of digital signature in the single-user setting are preserved in the multi-user setting [5,18], Dodis et al. showed that this is not necessarily true for optimistic fair exchange. They gave a counterexample that is secure in the former setting but insecure in the latter setting. A dishonest verifier succeeds in converting the signer's partial signature into a full one by colluding with the verifier in another transaction of exchange.

In a more recent work, [22] Huang et al. further improved Dodis et al.'s result by considering a more relaxed public key model called *chosen-key model* [26], in which the adversary is allowed to choose public keys arbitrarily without requiring to show its knowledge of the corresponding private keys. They showed that there is a gap between security of OFE in the chosen-key model and that in the *certified-key* model considered in [14] and previous works, in which the adversary has to prove its knowledge of the secret key before using a public key. They further proposed a generic construction of OFE in the chosen-key model without random oracles, which is based on a ring signature [29] and a standard signature [19].

A natural approach to constructing OFE is to use the arbitrator's public key to encrypt the signer's signature and then provide a non-interactive proof to show that the ciphertext indeed contains a valid signature of the signer. This is the well-known paradigm of constructing OFE from verifiably encrypted signature (VES) [9]. In general, we can always obtain such a scheme using NP-reduction. But efficiency is the issue preventing the resulting scheme from practical use. It is trivial to come up with a concrete and efficient instantiation of this paradigm secure in the standard model. To the best of our knowledge, the only known VES scheme secure without random oracles is due to Lu et al. [25], which is based on Waters signature scheme [32]. However, their scheme is merely proved to be secure in the single-user setting (and under the certified-key model).

(*Time Capsule Signature*). In FC 2005, Dodis and Yum [16] proposed the notion of time capsule signature. In a time capsule signature scheme, there is a semi-trusted time server, which honestly publishes the corresponding secret information at each time event t. Alice produces a "premature" signature σ' on a message m, which is claimed to become "mature" at time event t, and sends it to Bob, which verifies the validity of σ'. At time t, the time server publishes the secret information with respect to t, which can be used by anybody to convert σ' into a matured signature of Alice. Besides, Alice can also *pre-hatch* her signature σ' before the claimed time t.

Recently, Hu et al. [20] proposed an efficient time capsule signature scheme based on Waters signature [32], which in turn is based on Computational Diffie–Hellman (CDH) assumption. Their scheme is proved to be secure without random oracles. However, pre-hatched signatures are *distinguishable* from the hatched signatures, thus not satisfying the *ambiguity* (see Sect. 3.2 for definition). Libert and Quisquater [24] later proposed another time capsule signature scheme secure without random oracles, which is based on Waters signature [32] as well. Different from [20], their scheme satisfies the ambiguity.

1.3 Paper organization

We review the definitions and security models of optimistic fair exchange and time capsule signature in Sects. 2 and 3, respectively. Our generic construction of OFE is then proposed in Sect. 4. The security of it is analyzed in Sect. 5. In Sect. 6, we give a concrete instantiation of our construction, which is then compared with some existing OFE schemes secure in the multi-user setting in Sect. 7. The article is concluded in Sect. 8.

2 Definitions and security model

Let $k \in \mathbb{N}$ be a security parameter. If x is a binary string, $|x|$ denotes the length of x; if S is a set, $|S|$ denotes the cardinality of S. For any binary strings x and y, $x\|y$ denotes the concatenation of x and y. By $x \leftarrow S$, we denote the operation that process S is performed and the output is x if S is an algorithm, or that x is randomly and uniformly selected from S if it is a distribution. By $x := (a, b, c)$, we denote the simple assignment operation. By "PPT", we mean that an algorithm runs in probabilistic polynomial-time. A function f is said to be *negligible* in k, if for every positive polynomial $\mathrm{poly}(\cdot)$ and for all sufficiently large k, we have that $f(k) < 1/\mathrm{poly}(k)$.

2.1 Definitions in the multi-user setting

Our definition for non-interactive optimistic fair exchange (OFE) follows the one in the multi-user setting given in [14].

Definition 1 A non-interactive optimistic fair exchange (OFE) involves two users (a signer and a verifier) and an arbitrator and is formalized using the following PPT algorithms:

$\mathsf{Setup}^{\mathsf{TTP}}$. On input 1^k, it generates a secret arbitration key ASK and a public partial verification key APK.

$\mathsf{Setup}^{\mathsf{User}}$. On input 1^k and (optionally) APK, it outputs a secret/public key pair (SK, PK). For a user U_i, we use (SK_{U_i}, PK_{U_i}) to denote the user's key pair.

$\mathsf{Sig}/\mathsf{Ver}$. Similar to the signing and verification algorithms of an ordinary digital signature scheme, $\mathsf{Sig}(m, SK_{U_i}, APK)$ outputs a signature σ_{U_i}, while $\mathsf{Ver}(m, \sigma_{U_i}, PK_{U_i}, APK)$ outputs 1 for acceptance or 1 for rejection, where message m is chosen by user U_i from the message space $\mathcal{M}$ defined under PK_{U_i}.

$\mathsf{PSig}/\mathsf{PVer}$. They are partial signing and verification algorithms, respectively, where PSig together with Res (defined below) should be functionally equivalent to Sig. $\mathsf{PSig}(m, SK_{U_i}, APK)$ outputs a partial signature ξ_{U_i}, while $\mathsf{PVer}(m, \xi_{U_i}, PK_{U_i}, APK)$ outputs 1 for acceptance or 0 for rejection.

Res. This is the resolution algorithm. $\mathsf{Res}(m, \xi_{U_i}, ASK, PK_{U_i})$ outputs a signature σ_{U_i}, or $\perp$ indicating the failure of resolving a partial signature.

In a typical OFE protocol run, the signer U_i first generates the partial signature ξ_{U_i} using PSig and sends it to the verifier. The verifier then checks the partial signature using PVer and fulfills his obligation if PVer outputs 1. After which, the signer sends the full signature σ_{U_i} to complete the transaction. If no problem occurs, the arbitrator does not participate in the protocol. However, if the signer refuses to send σ_{U_i} at the end, the verifier will send ξ_{U_i} as well as a proof of fulfilling his obligation to the arbitrator. The arbitrator will generate σ_{U_i} using Res and sends it to the verifier if the proof sounds. Similar to previous definitions (e.g., [14,15]), the definition does not deal with the application-specific question of how the verifier proves to the arbitrator that he has fulfilled his obligation to the signer. However, unlike previous definitions, we do not assume the authenticity of public keys.

Remark 1 (**An Optional Input of** $\mathsf{Setup}^{\mathsf{User}}$). In the definition above, APK is an optional input of $\mathsf{Setup}^{\mathsf{User}}$. This allows the arbitrator and the users to share some common system parameters without getting involved in any interactive registration phase. The advantage is that the setup-free feature [35,36] can be ensured while having common system parameters shared across the entire system without having a dedicated system parameter generation algorithm defined. For schemes where the users and the arbitrator do not share any system parameter, APK can simply be removed from the input of $\mathsf{Setup}^{\mathsf{User}}$.

2.2 Security models

The *correctness* requires that for all security parameters $k \in \mathbb{N}$, $(ASK, APK) \leftarrow \mathsf{Setup}^{\mathsf{TTP}}(1^k)$, $(SK_{U_i}, PK_{U_i}) \leftarrow \mathsf{Setup}^{\mathsf{User}}(1^k, APK)$, let $\xi_{U_i} \leftarrow \mathsf{PSig}(m, SK_{U_i}, APK)$, each of the following equations holds with probability 1:

$$\mathsf{PVer}(m, \xi_{U_i}, PK_{U_i}, APK) = 1,$$
$$\mathsf{Ver}(m, \mathsf{Sig}(m, SK_{U_i}, APK), PK_{U_i}, APK) = 1, \text{ and}$$
$$\mathsf{Ver}(m, \mathsf{Res}(m, \xi_{U_i}, ASK, PK_{U_i}), PK_{U_i}, APK) = 1.$$

The *ambiguity* property requires that the distribution of full signatures generated by the signer should be (computationally) indistinguishable from that of full signatures resolved by the arbitrator on input valid partial signatures. Formally, denote by

$$\Sigma_0 \stackrel{\text{def}}{=} \{\mathsf{Sig}(m, SK_{U_i}, APK)\}_{m\in\{0,1\}^*}$$

and

$$\Sigma_1 \stackrel{\text{def}}{=} \{\mathsf{Res}(m, \mathsf{PSig}(m, SK_{U_i}, APK), ASK, PK_{U_i})\}_{m\in\{0,1\}^*}.$$

We also denote by $\Sigma_0(m)$ (resp. $\Sigma_1(m)$) the subspace of Σ_0 (resp. Σ_1) defined by $m \in \{0, 1\}^*$. For any probabilistic polynomial-time algorithm $\mathcal{D}$, the following probability should be negligibly close to $1/2$:

$$\Pr[m \leftarrow \{0, 1\}^*, b \leftarrow \{0, 1\}, \sigma \leftarrow \Sigma_b(m),$$
$$b' \leftarrow \mathcal{D}(PK_{U_i}, APK, m, \sigma) : b' = b].$$

Ambiguity is useful in applications of OFE, in which the signer does not want to let others know whether a signature is resolved by the arbitrator. For example, Alice and Bob execute an OFE protocol to sign a contract online, but due to the internet fault, Alice fails to return her full signature in time. Thus, Bob asks the arbitrator to resolve her signature. If the scheme is not extraction ambiguous, outsiders may think Alice is cheating, and this reduces the credit of Alice.

Readers may note that the definition of ambiguity above does not discuss if the adversary has any oracle access. In fact, similar to ring signature, we may specify various levels of ambiguity for OFE as well. They may include *basic ambiguity*, *ambiguity with respect to adversarially-chosen keys*, and *ambiguity against attribution attacks/full key exposure*. Readers may refer to [7] for their definitions in the context of ring signature. The ambiguity definition above follows that given in [14,15] with the sole purpose of making the construction of OFE non-trivial. For stronger notions, say *basic ambiguity*, we may require that no probabilistic polynomial-time adversary can distinguish full signatures generated by the signer from those resolved by the arbitrator with non-negligible advantage, even if the adversary can access partial signature oracle and resolution oracle.

The security of optimistic fair exchange consists of three aspects: *security against signers*, *security against verifiers*, and *security against the arbitrator*. The definitions of them in the multi-user setting are given as follows.

Security Against Signers. Intuitively, we require that no PPT adversary A should be able to produce a partial signature with non-negligible probability, which looks good to verifiers but cannot be resolved to a full signature by the honest arbitrator. This ensures the fairness for verifiers, that is, if the signer has committed to a message, the verifier will always be able to get the full commitment of the signer. Formally, we consider the following experiment:

$$\mathsf{Setup}^{\mathsf{TTP}}(1^k) \rightarrow (ASK, APK)$$
$$(m, \xi, PK^*) \leftarrow A^{O_{\mathsf{Res}}}(APK)$$
$$\sigma \leftarrow \mathsf{Res}(m, \xi, ASK, PK^*)$$
$$\text{Success of } A := [\mathsf{PVer}(m, \xi, PK^*, APK) = 1 \wedge \mathsf{Ver}(m, \sigma, PK^*, APK) = 0]$$

where oracle O_{Res} takes as input a valid partial signature ξ of user U_i on message m, i.e., (m, ξ, PK_{U_i}), and outputs a full signature σ on m under PK_{U_i}. The advantage of A in the experiment $\mathrm{Adv}_{\mathrm{A}}(k)$ is defined to be A's success probability.

Security Against Verifiers. This security notion requires that any PPT verifier B should not be able to transform a partial signature into a full signature with non-negligible probability if no help has been obtained from the signer or the arbitrator. This requirement has some similarity to the notion of *opacity* for verifiably encrypted signature [9]. Formally, we consider the following experiment:

$$\mathsf{Setup}^{\mathsf{TTP}}(1^k) \rightarrow (ASK, APK)$$
$$\mathsf{Setup}^{\mathsf{User}}(1^k) \rightarrow (SK, PK)$$
$$(m, \sigma) \leftarrow B^{O_{\mathsf{PSig}}, O_{\mathsf{Res}}}(PK, APK)$$
$$\text{Success of } B := [\mathsf{Ver}(m, \sigma, PK, APK) = 1 \wedge (m, \cdot, PK) \notin Query(B, O_{\mathsf{Res}})]$$

where oracle O_{Res} is described in the previous experiment, the partial signing oracle O_{PSig} takes as input a message m and returns a valid partial signature ξ on m under PK, and $Query(B, O_{\mathsf{Res}})$ is the set of valid queries B issued to the resolution oracle O_{Res}. The advantage of B in the experiment $\mathrm{Adv}_{\mathrm{B}}(k)$ is defined to be B's success probability.

Security Against the Arbitrator. Intuitively, this security notion requires that any PPT arbitrator C should not be able to generate with non-negligible probability a full signature without explicitly asking the signer for generating one. This ensures the fairness for signers, that is, no one can frame the actual signer on a message with a forgery. Formally, we consider the following experiment:

$$\mathsf{Setup}^{\mathsf{TTP}^*}(1^k) \rightarrow (ASK^*, APK)$$
$$\mathsf{Setup}^{\mathsf{User}}(1^k) \rightarrow (SK, PK)$$
$$(m, \sigma) \leftarrow C^{O_{\mathsf{PSig}}}(ASK^*, APK, PK)$$
$$\text{Success of } C := [\mathsf{Ver}(m, \sigma, PK, APK) = 1 \wedge (m, \cdot) \notin Query(C, O_{\mathsf{PSig}})]$$

where $\mathsf{Setup}^{\mathsf{TTP}^*}$ denotes the run of $\mathsf{Setup}^{\mathsf{TTP}}$ by a dishonest arbitrator (run by C), the partial signing oracle O_{PSig} is described in the previous experiment, ASK^* is C's state information, and $Query(C, O_{\mathsf{PSig}})$ is the set of queries C issued to the partial signing oracle O_{PSig}. The advantage of C in this experiment $\mathrm{Adv}_{\mathrm{C}}(k)$ is defined to be C's success probability.

Definition 2 A non-interactive optimistic fair exchange scheme is said to be *secure in the multi-user setting* if there is no PPT adversary that wins any of the experiments above with non-negligible advantage.

Since we are considering *certified-key* model in this work, in all the aforementioned experiments the adversary has to prove its knowledge of secret key for each public key it chooses and uses. Usually, we give the adversary access to an extra oracle, O_{kr}, which takes as input a key pair (PK, SK) and (stores SK and) returns PK if the pair is well-formed, i.e., (PK, SK) is a possible output of the key generation algorithm and $\perp$ otherwise. If we consider *chosen-key* model, the adversary does not have such a restriction during the attack.

2.3 On the multi-arbitrator setting

One may also notice that in the experiment for formalizing Security Against the Arbitrator, the adversary C has two phases. In the first phase, C merely generates APK without having access to O_{PSig}. In the second phase, C is to generate a forgery while allowing access to O_{PSig} with respect to APK. The purpose of having this two-phase arrangement is to make sure that the model is under the *single-arbitrator* setting. Although all the security requirements of optimistic fair exchange schemes are studied under the multi-user setting in this article, to be consistent with previous work [14,15], we restrict ourselves to focus on the formalization of a system which allows only one arbitrator. On the other side, if we combine the two phases in the experiment for Security Against the Arbitrator into one, that is, the second and the third statements are combined and replaced as follows,

$$(APK, m, \sigma) \leftarrow C^{O_{\mathsf{PSig}}}(PK)$$

and modify O_{PSig} by taking an additional input, which is a public partial verification key APK', then we are able to consider the *multi-arbitrator* setting for this security notion (by also changing the restriction such that we only require $(m, \cdot, APK) \notin Query(C, O_{\mathsf{PSig}})$).

2.4 On the validity of a partial signature

In optimistic fair exchange, a partial signature shows that Alice (the signer) is willing to exchange items with Bob (the

verifier). For example, in the schemes proposed in [14,22], the signer's partial signature is its signature on the message generated using a standard signature scheme. Due to its public verifiability (e.g., using PVer algorithm), Bob can show to anyone else Alice's will by releasing the partial signature it received from Alice, and possibly obtain benefit from other parties. Huang et al. addressed this issue, and proposed the notion of ambiguous optimistic fair exchange [21], in which it is required that partial signatures of Alice are indistinguishable from those of Bob. However, this is out of the scope of our work, as we focus on traditional type of optimistic fair exchange.

3 Time capsule signatures

3.1 Definition

Time capsule signature, introduced by Dodis and Yum in [16], is a kind of digit signature schemes which allows a signature to bear a (future) time t so that the signature will only become valid at time t or later, after a semi-trusted third party, called time server, releases time-dependent information. Besides, the real signer of a time capsule signature has the privilege to make a time capsule signature valid before time t.

Definition 3 ([16]) A *time capsule signature* scheme is specified by an 8-tuple of PPT algorithms ($\mathsf{Setup}^{\mathsf{TS}}$, $\mathsf{Setup}^{\mathsf{User}}$, TSig, TVer, TRelease, Hatch, PreHatch, Ver) such that:

$\mathsf{Setup}^{\mathsf{TS}}$. This setup algorithm is run by the Time Server. It takes a security parameter 1^k and returns a private/public time release key pair (TSK, TPK).

$\mathsf{Setup}^{\mathsf{User}}$. This setup algorithm is run by each user. It takes as input 1^k and returns the user's private/public key pair (SK, PK).

TSig. The time capsule signature generation algorithm TSig takes as input (m, SK, TPK, t) where $t \in \mathcal{T}$ is a specific time event from which the signature becomes valid, and outputs a time capsule signature ξ_t.

TVer. The time capsule signature verification algorithm TVer takes (m, ξ_t, PK, TPK, t) and returns 1 for acceptance or 0 for rejection.

TRelease. This time release algorithm TRelease takes as input (t, TSK). At the beginning of each time event t, the time server publishes $z_t \leftarrow \mathsf{TRelease}(t, TSK)$.

Hatch. This algorithm is run by any party and is used to open a valid time capsule signature which became mature. It takes as input (m, ξ_t, PK, TPK, z_t) and returns a hatch signature σ_t.

PreHatch. This algorithm is run by the signer and used to open a valid time capsule signature which is not mature yet. It takes as input (m, ξ_t, SK, TPK, t) and returns the pre-hatched signature σ_t.

Ver. This algorithm is used to verify a hatched or pre-hatched signature. Ver takes as input $(m, \sigma_t, PK, TPK, t)$ and returns 1 for acceptance or 0 for rejection.

The time server runs $\mathsf{Setup}^{\mathsf{TS}}$ to generate its key pair, and each user runs $\mathsf{Setup}^{\mathsf{User}}$ to generate a user key pair. Any user can run TSig to produce a time capsule signature to be valid at some time period t and can later make its signature mature before t by running PreHatch. At each time period t, the time servers runs TRelease to release some secret information related to t with which any user can make a time capsule signature mature and verify its validity.

3.2 Security models

The *correctness* requirement states that for any $m \in \{0,1\}^*$ and $t \in \mathcal{T}$, let $\xi_t \leftarrow \mathsf{TSig}(m, SK, TPK, t)$ and $z_t \leftarrow \mathsf{TRelease}(t, TSK)$, each of the following equations holds with probability 1:

$$\mathsf{TVer}(m, \xi_t, PK, TPK, t) = 1,$$

$$\mathsf{Ver}(m, \mathsf{Hatch}(m, \xi_t, PK, TPK, z_t), PK, TPK, t) = 1,$$

and

$$\mathsf{Ver}(m, \mathsf{PreHatch}(m, \xi_t, SK, TPK, t), PK, TPK, t) = 1.$$

The *ambiguity* property requires that the "hatched signature" $\tilde{\sigma}_t$ is (computationally) *indistinguishable* from the "pre-hatched signature" σ_t, even if the distinguisher knows TSK.

The security of time capsule signatures consists of three aspects: security against the signer Alice, security against the verifier Bob and security against time server. In the following, we denote by O_{TSig} the oracle simulating the algorithm TSig, which takes (m, t) as input and returns Alice's time capsule signature ξ_t, by O_{TR} the time release oracle, which takes t as input and returns the secret time information z_t, and by O_{PreH} the oracle simulating algorithm PreHatch, which takes (m, t, ξ_t) as input and returns Alices' pre-hatch signature σ.

Security Against Alice. We require that any PPT adversary A could succeed with at most negligible probability in the following experiment:

$$\mathsf{Setup}^{\mathsf{TS}} - (1^k) \rightarrow (TSK, TPK)$$

$$(m, t, \xi_t, PK) \leftarrow A^{O_{\mathsf{TR}}}(TPK)$$

$$z_t \leftarrow \mathsf{TRelease}(t, TSK)$$

$$\sigma_t \leftarrow \mathsf{Hatch}(m, \xi_t, PK, TPK, z_t)$$

$$\text{Success of } A := [\mathsf{TVer}(m, \xi_t, PK, TPK, t) = 1 \wedge \mathsf{Ver}(m, \sigma_t, PK, TPK, t) = 0]$$

Security Against Bob. We require that any PPT adversary B could succeed with at most negligible probability in the following experiment:

$\mathsf{Setup}^{\mathsf{TS}} - (1^k) \rightarrow (TSK, TPK)$
$\mathsf{Setup}^{\mathsf{User}} - (1^k) \rightarrow (SK, PK)$
$(m, t, \sigma_t) \leftarrow B^{O_{\mathsf{TSig}}, O_{\mathsf{TR}}, O_{\mathsf{PreH}}}(PK, TPK)$
Success of $B := [\mathsf{Ver}(m, \sigma_t, PK, TPK, t) = 1 \wedge t \notin Query(B, O_{\mathsf{TR}}) \wedge (m, t, \cdot) \notin Query(B, O_{\mathsf{PreH}})]$

where $Query(B, O_{\mathsf{TR}})$ is the set of queries B issued to the time release oracle O_{TR}, and $Query(B, O_{\mathsf{PreH}})$ is the set of *valid* queries B issued to the pre-hatch oracle O_{PreH}, i.e., (m, t, ξ_t) such that $\mathsf{TVer}(m, \xi_t, PK, TPK, t) = 1$.

Security Against Time Server. We require that any PPT adversary C could succeed with at most negligible probability in the following experiment:

$\mathsf{Setup}^{\mathsf{TS}^*} - (1^k) \rightarrow (TSK^*, TPK)$
$\mathsf{Setup}^{\mathsf{User}} - (1^k) \rightarrow (SK, PK)$
$(m, t, \sigma_t) \leftarrow C^{O_{\mathsf{TSig}}, O_{\mathsf{PreH}}}(PK, TPK, TSK^*)$
Success of $C := [\mathsf{Ver}(m, \sigma_t, PK, TPK, t) = 1 \wedge (m, \cdot) \notin Query(C, O_{\mathsf{TSig}})]$

where $\mathsf{Setup}^{\mathsf{TS}^*}$ denotes the run of $\mathsf{Setup}^{\mathsf{TS}}$ with a dishonest time server (run by C), TSK^* is C's state after this run, and $Query(C, O_{\mathsf{TSig}})$ is the set of queries C issued to the time capsule signature generation oracle O_{TSig} with the restriction that $(m, t') \notin Query(C, O_{\mathsf{TSig}})$ for all $t' \in \mathcal{T}$.

4 Our optimistic fair exchange scheme

In this section, we will show another way of constructing OFE schemes secure in the multi-user setting and certified-key model. Let $\mathsf{TCS} = (\mathsf{Setup}^{\mathsf{TS}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{TSig}, \mathsf{TVer}, \mathsf{TRelease}, \mathsf{Hatch}, \mathsf{PreHatch}, \mathsf{Ver})$ be a time capsule signature scheme. In the following, we show how to use TCS to build an optimistic fair exchange scheme OFE' secure in the multi-user setting and the certified-key model.

Let k be the security parameter. Suppose that $\mathsf{H} : \{0, 1\}^* \rightarrow \mathcal{T}$ is a collision-free hash function, where $\mathcal{T}$ is the space of time events. Without loss of generality, we assume that the size of $\mathcal{T}$ is super-polynomial in k. This is to ensure the collision-freeness of H.

$\mathsf{Setup}^{\mathsf{TTP}}$. The arbitrator runs $\mathsf{TCS.Setup}^{\mathsf{TS}}(1^k)$ to generate a key pair (TSK, TPK), and sets $(ASK, APK) := (TSK, TPK)$.

$\mathsf{Setup}^{\mathsf{User}}$. Each user U_i generates a public/private key pair by computing $(SK_{U_i}, PK_{U_i}) \leftarrow \mathsf{TCS.Setup}^{\mathsf{User}}(1^k)$.

Sig. On input a message m, the signer U_i generates a time event t [1] by computing

$$t \leftarrow \mathsf{H}(m, PK_{U_i}).$$

It then computes the full signature as

$$\sigma \leftarrow \mathsf{TCS.PreHatch}(m, \xi, SK_{U_i}, APK, t),$$

where $\xi \leftarrow \mathsf{TCS.TSig}(m, SK_{U_i}, APK, t)$.

Ver. On input a message m and a signature σ purportedly produced by U_i, the verifier computes $t \leftarrow \mathsf{H}(m, PK_{U_i})$ and returns

$$\mathsf{TCS.Ver}(m, \sigma, PK_{U_i}, APK, t).$$

PSig. On input a message m, the signer U_i computes

$$t \leftarrow \mathsf{H}(m, PK_{U_i}), \quad \xi \leftarrow \mathsf{TCS.TSig}(m, SK_{U_i}, APK, t).$$

It returns ξ.

PVer. On input a message m and a partial signature ξ purportedly produced by U_i, the verifier computes

$$t \leftarrow \mathsf{H}(m, PK_{U_i}), \quad b \leftarrow \mathsf{TCS.TVer}(m, \xi, PK_{U_i}, APK, t),$$

and returns the bit b.

Res. On input a message m and a partial signature ξ of user U_i, the arbitrator first checks if ξ is a valid signature on m with respect to PK_{U_i}. If not, it rejects the input by outputting $\perp$; otherwise, it computes

$$t \leftarrow \mathsf{H}(m, PK_{U_i}), \quad z_t \leftarrow \mathsf{TCS.TRelease}(t, ASK),$$

and

$$\sigma \leftarrow \mathsf{TCS.Hatch}(m, \xi, PK_{U_i}, APK, z_t).$$

The arbitrator returns σ.

This construction is *setup-free*. The *stand-alone* property depends on that of the underlying time capsule signature. The correctness of OFE' is obvious and the ambiguity property simply follows that of TCS.

Remark 2 (**On the Space $\mathcal{T}$ of Time Events**) As of our best knowledge, all the time capsule signature schemes in

[1] The reason of computing t rather than randomly selecting t is to ensure that in the generation of each signature, the time event is distinct if the message or the signer is different, which is important in the proof of security against verifiers. To be shown later, as in the proof of Lemma.

the literature [16,20,24][2] put no restriction/limitation on the range of possible time events. In fact, the time event t in these schemes can take any values from $\{0,1\}^*$, since a mechanism analogous to identity-based cryptography is used in their constructions and t behaves as an identity. Therefore, it is reasonable for us to assume that the size of $\mathcal{T}$ is at least super-polynomial in the security parameter, or large enough for guaranteeing the collision-resistance of H. Besides, if the time event t can take any arbitrary value (i.e., $\{0,1\}^*$), then we can simply remove H in our construction above for reducing the basic assumption for building OFE'. That is, we directly use $m \| PK_{U_i}$ instead of the hashed value of it as the "time event" t.

5 Security analysis

For the security of the above construction of OFE, we have the following theorem. Note that since the security of time capsule signatures is defined in a compatible and very similar way to that of OFE in [14], in the following, we only show the security of OFE' in the certified-key model.

Theorem 1 *If there exist secure time capsule signature schemes and collision-free hash functions, there exist secure optimistic fair exchange schemes in the multi-user setting and the certified-key model.*

The theorem follows Lemma 1 (security against signers, Lemma 2 (security against verifiers) and Lemma 3 (security against the arbitrator).

Lemma 1 *The optimistic fair exchange scheme* OFE' *above is secure against signers.*

Proof Suppose that A is a PPT adversary that breaks the security against signers of OFE' with non-negligible advantage ϵ_A. We construct a PPT algorithm $\bar{A}$ which breaks the security against the signer of TCS.

Given the time server public key TPK and a time release oracle O_{TR} which simulates the $\mathsf{TCS.TRelease}$ algorithm, $\bar{A}$ randomly selects a hash function $\mathsf{H} : \{0,1\}^* \to \mathcal{T}$, and runs A on input (TPK, H). During the execution, A has access to oracle O_{Res}. To answer A's query (m, ξ, PK_{U_i}), $\bar{A}$ first checks the validity of ξ by running $\mathsf{OFE}'.\mathsf{PVer}(m, \xi, PK_{U_i}, APK)$. If invalid, $\bar{A}$ returns $\perp$. Otherwise, it issues a query to its oracle O_{TR} on input $t \leftarrow \mathsf{H}(m, PK_{U_i})$, which returns the corresponding z_t. $\bar{A}$ then computes $\sigma \leftarrow \mathsf{TCS.Hatch}(m, \xi, PK_{U_i}, TPK, z_t)$ and returns σ back to A. Note that the above simulation of O_{Res} is perfect.

Finally, A outputs (m, ξ, PK). Without loss of generality, we assume that A wins the game. This happens with probability ϵ_A. (If A fails, $\bar{A}$ also fails and halts.) Thus, we get that $\mathsf{OFE}'.\mathsf{PVer}(m, \xi, PK, TPK) = 1$ and $\mathsf{OFE}'.\mathsf{Ver}(m, \sigma, PK, TPK) = 0$, where $\sigma \leftarrow \mathsf{OFE}'.\mathsf{Res}(m, \xi, ASK, PK)$. This indicates that $\mathsf{TCS}.\mathsf{TVer}(m, \xi, PK, TPK, t) = 1$ and $\mathsf{TCS}.\mathsf{Ver}(m, \sigma, PK, TPK, t) = 0$, where $t \leftarrow \mathsf{H}(m, PK)$. Hence, we let $\bar{A}$ output (m, t, ξ, PK), and $\bar{A}$ wins its game with probability ϵ_A. □

Remark 3 Note that in the proof, after receiving the output (m, ξ, PK) of A, $\bar{A}$ can actually compute σ by generating the time event t as described above, issuing a query to oracle O_{TR} to get z_t, and then running $\sigma_t \leftarrow \mathsf{TCS.Hatch}(m, \xi_t, PK_A, TPK, z_t)$. If t was ever issued by $\bar{A}$ to O_{TR} during the simulation, $\bar{A}$ can simply retrieve the corresponding z_t from its memory instead of issuing a new query. Therefore, $\bar{A}$ can check the validity of A's output and decides to output (m, t, ξ, PK_A) or to abort.

Lemma 2 *The optimistic fair exchange scheme* OFE' *above is secure against verifiers.*

Proof Suppose that B is a PPT adversary which breaks the security against verifiers of OFE' with non-negligible advantage ϵ_B, we construct a PPT algorithm $\bar{B}$ which breaks the security against the verifier of TCS.

Given the time server public key TPK, the signer's public key PK, and oracles O_{TSig} simulating algorithm $\mathsf{TCS.TSig}$, O_{TR} simulating algorithm $\mathsf{TCS.TRelease}$ and O_{PreH} simulating algorithm $\mathsf{TCS.PreHatch}$, $\bar{B}$ randomly selects a hash function $\mathsf{H} : \{0,1\}^* \to \mathcal{T}$, and runs B on input (TPK, PK, H). To simulate oracles O_{PSig} and O_{Res} for B, $\bar{B}$ uses O_{TSig} and O_{TR} respectively, as follows.

- When B issues a query to O_{PSig} on input m, $\bar{B}$ generates time event $t \leftarrow \mathsf{H}(m, PK)$, and issues a query to its oracle O_{TSig} on input (m, t), which returns the signer's signature ξ_t. $\bar{B}$ then returns ξ to B.
- When B issues a *valid* query to O_{Res} on input (m, ξ, PK_{U_i}), $\bar{B}$ generates time event $t \leftarrow \mathsf{H}(m, PK_{U_i})$, and issues a query to O_{TR} on input t which returns the corresponding z_t. $\bar{B}$ then returns $\sigma \leftarrow \mathsf{TCS.Hatch}(m, \xi, PK_{U_i}, TPK, z_t)$.

It is readily seen that the above simulation is perfect. Finally, B outputs (m, σ). Without loss of generality, we assume that B wins the game. Thus, we have that $\mathsf{OFE}'.\mathsf{Ver}(m, \sigma, PK, TPK) = 1$ and $(m, \cdot, PK) \notin Query(B, O_{\mathsf{Res}})$. Since the hash function H is collision-free, it holds with only negligible probability that $t \leftarrow \mathsf{H}(m, PK)$ is the same as one of the previous time events generated by $\bar{B}$ during the simulation of O_{Res} and O_{PSig}. Otherwise, B and

[2] We note that schemes in [20] are not ambiguous. That is, the pre-hatched signatures are distinguishable from hatched signatures.

$\bar{B}$ together form an algorithm breaking the collision-freeness property of H. It is well understood that if t appeared before, $\bar{B}$ fails and halts. So we have that $\bar{B}$ did not issue a query to O_{TR} on input t. Also note that during the whole execution, $\bar{B}$ never issued a query to O_{PreH}. Therefore, we can let $\bar{B}$ output (m, t, σ) and $\bar{B}$ succeeds in its game with probability $\epsilon_{\bar{B}}$ so $\left|\epsilon_B - \epsilon_{\bar{B}}\right|$ is negligible in k. The difference is due to the negligible probability that a collision of H occurs. □

Lemma 3 *The optimistic fair exchange scheme* OFE' *above is secure against the arbitrator.*

Proof Suppose that C is a PPT adversary which breaks the security against the arbitrator of OFE' with non-negligible advantage ϵ_C, we construct a PPT algorithm $\bar{C}$ which breaks the security against the time server of TCS.

Given the time server private/public key pair (TSK^*, TPK), the public key PK of the signer Alice, and oracles O_{TSig} simulating algorithm $\mathsf{TCS.TSig}$, and O_{PreH} simulating algorithm $\mathsf{TCS.PreHatch}$, $\bar{C}$ randomly selects a hash function $\mathsf{H} : \{0,1\}^* \to \mathcal{T}$, and runs C on input $(TSK^*, PK, TPK, \mathsf{H})$. To simulate the oracle O_{PSig} for C, $\bar{C}$ generates the time event t as described in the $\mathsf{OFE}'.\mathsf{PSig}$ algorithm, and then issues a query to O_{TSig} on input (m, t), which returns Alice's time capsule signature ξ. $\bar{C}$ returns ξ to C. It's easy to see that the simulation is perfect.

Finally, C outputs (m, σ). Again, we simply assume C wins its game. This happens with probability ϵ_C. Thus, we have that $\mathsf{OFE}'.\mathsf{Ver}(m, \sigma, PK, TPK) = 1$ and $m \notin Query(C, O_{\mathsf{PSig}})$. It indicates that $\bar{C}$ didn't issue a query to its oracle O_{TSig} on input (m, t') for any t'. Also note that during the simulation, $\bar{B}$ never issued a query to its oracle O_{PreH}. Therefore, we can let $\bar{C}$ output (m, t, σ) where $t \leftarrow \mathsf{H}(m, PK)$, and $\bar{B}$ succeeds in its game with probability ϵ_C. □

6 An instantiation without random oracles

Recently, Libert and Quisquater [24] proposed an efficient time capsule signature scheme proven secure in the standard model based on Waters signature [32], which in turn is based on Computational Diffie–Hellman (CDH) assumption. By instantiating our generic construction above using their time capsule signature scheme, the final scheme will also enjoy the security without random oracles. In this particular scheme, we even do not need to introduce another collision-resistant hash function either. This is because the collision-free hash function $H : \{0,1\}^* \to \{0,1\}^n$ has already been employed in Libert-Quisquater time capsule signature scheme. We can simply use H to map $m \| PK_{U_i}$ into the time event space $\{0,1\}^n$, which is exactly the case in their implementation.

Let $\mathbb{G}, \mathbb{G}_T$ be two cyclic and multiplicative groups of prime order p, and g be a random generator of $\mathbb{G}$. Let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing. Let $H : \{0,1\}^* \to \{0,1\}^n$ be a collision-resistant hash function. The concrete OFE scheme works as below.

$\mathsf{Setup}^{\mathsf{TTP}}$. The arbitrator chooses $g_2 \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, and computes $g_1 = g^\alpha$ and $V = \hat{e}(g_1, g_2)$. It then chooses a random vector $\mathbf{v} = (v', v_1, \cdots, v_n) \in \mathbb{G}^{n+1}$ defining a function $F_v : \{0,1\}^n \to \mathbb{G}$ such that for any $t \in \{0,1\}^n$, $F_v(t) = v' \prod_{j=1}^{n} v_j^{t_j}$. The private key of the arbitrator is $ASK = g_2^\alpha$ and the public key is $APK = \{n, g_1, g_2, \mathbf{v}, V\}$.

$\mathsf{Setup}^{\mathsf{User}}$. The signer selects $\beta \in \mathbb{Z}_p$, $h \in \mathbb{G}$ and a random $(n+1)$-vector $\mathbf{u} = (u', u_1, \cdots, u_n) \in \mathbb{G}^{n+1}$ defining a function $F_u : \{0,1\}^n \to \mathbb{G}$ such that for any $m \in \{0,1\}^n$, $F_u(m) = u' \prod_{j=1}^{n} u_j^{m_j}$. Its private key is $SK = h^\beta$, and public key is $PK = \{h, \hat{h}, \mathbf{u}, U\}$, where $\hat{h} = g^\beta$ and $U = \hat{e}(h, \hat{h})$.

PSig. Given a message m, the signer U_i chooses $j_1, j_2 \leftarrow \mathbb{Z}_p$ and computes $c = g_2^{j_1} g^{j_2}$ and $t \leftarrow H(m, PK_{U_i})$. It then picks $r, s \leftarrow \mathbb{Z}_p$, computes $d_1 = c^s g_1^{-j_2/j_1} F_v(t)^r$, sets $d_2 = g^r$ and $d_3 = g^s g_1^{-1/j_1}$. The signer also computes $M \leftarrow H(m \| c \| t) \in \{0,1\}^n$ and

$$(\xi_1, \xi_2) = (h^\beta F_u(M)^{\hat{r}}, g^{\hat{r}})$$

for a randomly chosen $\hat{r} \leftarrow \mathbb{Z}_p$. It outputs the partial signature $\xi = (\xi_1, \xi_2, c)$, and stores (d_1, d_2, d_3).

PVer. Given $(m, \xi = (\xi_1, \xi_2, c))$ purportedly produced by U_i, the verifier computes $t \leftarrow H(m, PK_{U_i})$ and $M \leftarrow H(m \| c \| t)$, and checks if $c \in \mathbb{G}$ and

$$\hat{e}(\xi_1, g) = U \cdot \hat{e}(F_u(M), \xi_2).$$

It outputs 1 if both hold, and 0 otherwise.

Sig. To fully sign a message m, the signer U_i sets $t \leftarrow H(m, PK_{U_i})$ and computes (ξ_1, ξ_2, c) and (d_1, d_2, d_3) as in the partial signing algorithm. It outputs the full signature $\sigma = (\xi_1, \xi_2, c, d_1, d_2, d_3)$.

Ver. Given $(m, \sigma = (\xi_1, \xi_2, c, d_1, d_2, d_3)$ purportedly produced by U_i, the verifier computes $t \leftarrow H(m, PK_{U_i})$ and $M \leftarrow H(m \| c \| t)$. It outputs 1 if

$$\hat{e}(d_1, g) = V \cdot \hat{e}(F_v(t), d_2) \cdot \hat{e}(c, d_3),$$
$$\hat{e}(\xi_1, g) = U \cdot \hat{e}(F_u(M), \xi_2),$$

and 0 otherwise.

Res. To resolve U_i's partial signature $\xi = (\xi_1, \xi_2, c)$ on message m, the arbitrators returns $\perp$ if ξ is not valid. Otherwise, it picks random $\tilde{r}, s \leftarrow \mathbb{Z}_p$ and computes $t \leftarrow H(m, PK_{U_i})$ and

$$(\tilde{d}_1, \tilde{d}_2, \tilde{d}_3) = (g_2^\alpha F_v(t)^{\tilde{r}} c^s, g^{\tilde{r}}, g^s).$$

Table 2 A detailed comparison with some existing results in the multi-user setting

	[14]	[22] **Inst 1**	[22] **Inst 2**	[22] **Inst 3**	Ours
Apk	$\|n\| + 1\mathbb{Z}_n$	$1\mathbb{G}$	$1\mathbb{G}$	$3\mathbb{G}$	$(k+3)\mathbb{G} + 1\mathbb{G}_T$
Pk	$1\mathbb{Z}_p$	$(k+3)\mathbb{G}$	$3\mathbb{G}$	$3\mathbb{G}$	$(k+3)\mathbb{G} + 1\mathbb{G}_T$
PSig	$3t + 2\mathbb{Z}_q + 1\mathbb{Z}_n^*$	$2\mathbb{G}$	$1\mathbb{G} + 1\mathbb{Z}_{n'}$	$2\mathbb{G} + 2\mathbb{Z}_p$	$3\mathbb{G}$
Sig	$2t + 1\mathbb{Z}_q + 1\mathbb{Z}_n^*$	$8\mathbb{G}$	$12\mathbb{G} + 5\mathbb{Z}_{n'}$	$5\mathbb{G} + 5\mathbb{Z}_p$	$6\mathbb{G}$
Crs	–	$(k+4)\mathbb{G}$	$2\mathbb{G}$	$4\mathbb{G}$	–
Assump	RSA + DL	CDH	SDH + SGD	Poly-SDH	CDH
Model	ROM	CRS	CRS	CRS	STD

k is the security parameter
A '–' in the row of 'Crs' means that the scheme does not impose a common reference string, except standard system parameters, e.g. group description and generator
Please refer to Table 1 notes

It outputs $\sigma = (\xi_1, \xi_2, c, \tilde{d}_1, \tilde{d}_2, \tilde{d}_3)$.

Libert et al. proved in [24] that their time capsule signature scheme is ambiguous and secure based on CDH assumption under the model given in Sect. 3.2. Combining their result and Theorem 1, we obtain the following corollary immediately.

Corollary 1 *The OFE scheme above is secure without random oracles provided that the hash function H is collision-resistant and CDH assumption holds.*

7 Comparison

Table 2 shows the comparison of our scheme with some existing results on OFE in terms of key size, signature size, length of common reference string, underlying assumptions and the need of random oracle model. As we consider multi-user setting in this work, we select those schemes proved to be secure in the multi-user setting for comparison in Table 2.

From the comparison, we can see that our scheme outperforms others in terms of signature size. Besides, the security of our scheme relies on the weakest number-theoretic assumption and is proved in the standard model, while the security of other schemes are based on stronger assumptions, and are proved in either random oracle model or common reference model. However, our scheme is based on Waters signature, and thus inherits its drawback. That is, our scheme suffers from long public keys as well.

8 Conclusion

In this article, we observed that due to the very similar nature with optimistic fair exchange, it is straightforward to build an optimistic fair exchange scheme in the multi-user setting and the certified-key model from a time capsule signature scheme secure in the certified-key model in conjunction with a collision-resistant hash function. Combining recent work on time capsule signatures in the standard model and our generic transformation, we come up with an efficient optimistic fair exchange scheme secure without random oracles.

References

1. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: ACM Conference on Computer and Communications Security, pp. 7–17. ACM (1997)
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures (extended abstract). In: Advances in Cryptology—EUROCRYPT 98, Lecture Notes in Computer Science, vol. 1403, pp. 591–606. Springer, Berlin (1998)
3. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. IEEE J. Sel. Areas Commun. **18**(4), 593–610 (2000)
4. Bao, F., Wang, G., Zhou, J., Zhu, H.: Analysis and improvement of Micali's fair contract signing protocol. In: Proceedings of 9th Australasian Conference on Information Security and Privacy, ACISP 2004, Lecture Notes in Computer Science, vol. 3108, pp. 176–187. Springer, Berlin (2004)
5. Bellare, M., Boldyreva, A., Micali, S.: Public-key encryption in a multi-user setting: security proofs and improvements. In: Advances in Cryptology—EUROCRYPT 2000, Lecture Notes in Computer Science, vol. 1807, pp. 259–274. Springer, Berlin (2000)
6. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73. ACM (1993)
7. Bender, A., Katz, J., Morselli, R.: Ring signatures: stronger definitions, and constructions without random oracles. In: Proceedings of 3rd IACR Theory of Cryptography Conference, TCC 2006, Lecture Notes in Computer Science, vol. 3876, pp. 60–79. Springer, Berlin. Also at Cryptology ePrint Archive, Report 2005/304 (2006)
8. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Advances in Cryptology—EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3027, pp. 56–73. Springer, Berlin (2004)
9. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Advances in Cryptology—EUROCRYPT 2003, Lecture Notes in Computer Science, vol. 2656, pp. 416–432. Springer, Berlin (2003)
10. Boyen, X.: Mesh signatures: how to leak a secret with unwitting and unwilling participants. In: Advances in Cryptology—EURO-

CRYPT 2007, Lecture Notes in Computer Science, vol. 4515, pp. 210–227. Springer, Berlin (2007)
11. Camenisch, J., Damgård, I.: Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In: Advances in Cryptology—ASIACRYPT 2000, Lecture Notes in Computer Science, vol. 1976, pp. 331–345. Springer, Berlin (2000)
12. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. In: Proceedings of 30th ACM Symposium on Theory of Computing, pp. 209–218. ACM (1998)
13. Chandran, N., Groth, J., Sahai, A.: Ring signatures of sub-linear size without random oracles. In: Proceedings of 34th International Colloquium on Automata, Languages and Programming, ICALP 2007, Lecture Notes in Computer Science, vol. 4596, pp. 423–434. Springer, (2007)
14. Dodis, Y., Lee, P.J., Yum, D.H.: Optimistic fair exchange in a multi-user setting. In: Proceedings of Public Key Cryptography 2007, Lecture Notes in Computer Science, vol. 4450, pp. 118–133. Springer, Berlin. Also at Cryptology ePrint Archive, Report 2007/182 (2007)
15. Dodis Y., Reyzin, L.: Breaking and repairing optimistic fair exchange from PODC 2003. In: ACM Workshop on Digital Rights Management, DRM 2003, pp. 47–54. ACM (2003)
16. Dodis, Y., Yum, D.H.: Time capsule signatures. In: Proceedings of Financial Cryptography and Data Security 2005, Lecture Notes in Computer Science, vol. 3570, pp. 57–71. Springer, Berlin (2005)
17. Ezhilchelvan, P.D., Shrivastava, S.K.: A family of trusted third party based fair-exchange protocols. IEEE Trans. Dependable Secure Comput. **2**(4), 273–286 (2005)
18. Galbraith, S., Malone-Lee, J., Smart, N.: Public key signatures in the multi-user setting. Inf. Process. Lett. **83**(5), 263–266 (2002)
19. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen-message attack. SIAM J. Comput. **17**(2), 281–308 (1988)
20. Hu, B.C., Wong, D.S., Huang, Q., Yang, G., Deng, X.: Time capsule signature: Efficient and provably secure constructions. In: Proceedings of 4th European PKI Workshop: Theory and Practice, EuroPKI 2007, Lecture Notes in Computer Science, vol. 4582, pp. 126–142. Springer, Berlin. Full paper is available at Cryptology ePrint Archive, Report 2007/146 (2007)
21. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Advances in Cryptology—ASIACRYPT 2008, Lecture Notes in Computer Science, vol. 5350, pp. 74–89. Springer, Berlin (2008)
22. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In: Proceedings of Topics in Cryptology—CT-RSA 2008, Lecture Notes in Computer Science, vol. 4964, pp. 106–120. Springer, Berlin (2008)
23. Kremer, S.: Formal Analysis of Optimistic Fair Exchange Protocols. PhD thesis, Université Libre de Bruxelles (2003)
24. Libert, B., Quisquater, J.-J.: Practical time capsule signatures in the standard model from bilinear maps. In: Proceedings of 1st International Conference on Pairing-Based Cryptography, Pairing 2007, Lecture Notes in Computer Science, vol. 4575, pp. 23–38. Springer, Berlin (2007)
25. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential aggregate signatures and multisignatures without random oracles. In: Advances in Cryptology—EUROCRYPT 2006, Lecture Notes in Computer Science, vol. 4004, pp. 465–485. Springer, Berlin (2006)
26. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential aggregate signatures from trapdoor permutations. In: Advances in Cryptology—EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3027, pp. 74–90. Springer, Berlin (2004)
27. Micali, S.: Simple and fast optimistic protocols for fair electronic exchange. In: ACM Symposium on Principles of Distributed Computing, PODC 2003, pp. 12–19. ACM (2003)
28. Park, J.M., Chong, E.K., Siegel, H.J.: Constructing fair-exchange protocols for e-commerce via distributed computation of RSA signatures. In: PODC 2003, pp. 172–181. ACM (2003)
29. Rivest, R., Shamir, A., Tauman, Y.: How to leak a secret. In: Advances in Cryptology—ASIACRYPT 2001, Lecture Notes in Computer Science, vol. 2248, pp. 552–565. Springer, Berlin (2001)
30. Shacham, H., Waters, B.: Efficient ring signatures without random oracles. In: Proceedings of Public Key Cryptography 2007, Lecture Notes in Computer Science, vol. 4450, pp. 166–180. Springer, Berlin (2007)
31. Wang, G.: An abuse-free fair contract signing protocol based on the RSA signature. In: Proceedings of 14th International Conference on World Wide Web, WWW 2005, pp. 412–421. ACM (2005)
32. Waters, B.: Efficient identity-based encryption without random oracles. In: Advances in Cryptology—EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 114–127. Springer, Berlin (2005)
33. Zhang, Z., Zhou, Y., Feng, D.: Efficient and optimistic fair exchanges based on standard RSA with provable security. Cryptology ePrint Archive, Report 2003/178 (2004)
34. Zhu, H.: Constructing optimistic fair exchange protocols from committed signatures. Cryptology ePrint Archive, Report 2005/012 (2003)
35. Zhu, H., Bao, F.: Stand-alone and setup-free verifiably committed signatures. In: Proceedings of Topics in Cryptology—CT-RSA 2006, Lecture Notes in Computer Science, vol. 3860, pp. 159–173. Springer, Berlin (2006)
36. Zhu, H., Susilo, W., Mu, Y.: Multi-party stand-alone and setup-free verifiably committed signatures. In: Proceedings of Public Key Cryptography 2007, Lecture Notes in Computer Science, vol. 4450, pp. 134–149. Springer, Berlin (2007)