6-2011

Preserving transparency and accountability in optimistic fair exchange of digital signatures

Xinyi HUANG

Yi Mu

Willy Susilo

Jianying Zhou

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Citation

HUANG, Xinyi; Mu, Yi; Susilo, Willy; Zhou, Jianying; and DENG, Robert H.. Preserving transparency and accountability in optimistic fair exchange of digital signatures. (2011). *IEEE Transactions on Information Forensics and Security*. 6, (2), 489-512.
Available at: https://ink.library.smu.edu.sg/sis_research/1369

Published in IEEE Transactions on Information Forensics and Security, 2011 June, Volume 6, Issue 2, Pages 498-512
https://doi.org/10.1109/TIFS.2011.2109952

Preserving Transparency and Accountability in Optimistic Fair Exchange of Digital Signatures

Xinyi Huang, Yi Mu, *Senior Member, IEEE*, Willy Susilo, *Senior Member, IEEE*, Wei Wu, Jianying Zhou, and Robert H. Deng, *Senior Member, IEEE*

***Abstract*—Optimistic fair exchange (OFE) protocols are useful tools for two participants to fairly exchange items with the aid of a third party who is only involved if needed. A widely accepted requirement is that the third party's involvement in the exchange must be *transparent*, to protect privacy and avoid bad publicity. At the same time, a dishonest third party would compromise the fairness of the exchange and the third party thus must be responsible for its behaviors. This is achieved in OFE protocols with another property called *accountability*. It is unfortunate that the accountability has never been formally studied in OFE since its introduction ten years ago. In this paper, we fill these gaps by giving the first complete definition of accountability in OFE where one of the exchanged items is a digital signature and a generic (also the first) design of OFE where transparency and accountability coexist.**

***Index Terms*—Accountability, fair exchange, generic design, transparency.**

I. Introduction

THE objective of a fair exchange protocol is to allow two parties to fairly exchange their items so that no one can gain any advantage. A straightforward solution is to introduce a trusted third party as a mediator: each party sends the item to the trusted third party, who upon verifying the correctness of both items, forwards each item to the other party. Such protocols require that the third party be always online, as each exchange needs the assistance from the third party. In order to alleviate the requirement of an always-online third party, Asokan *et al.* [1] introduced the notion of *optimistic fair exchange* (hereinafter referred to as "OFE"). OFE also makes use of a third party (called the *arbitrator*), but it does not need to be always online; instead, the arbitrator only gets involved if something goes wrong (e.g., one party attempts to cheat or other faults occur).

A typical example of OFE is as follows: Two participants (say, Alice and Bob) first agree on the items to be exchanged: Alice's item is a valid $\mathsf{full\ signature}$ (e.g., the signature on a credit card purchase) and Bob's item is denoted by item_B (e.g., a book). At the end of the exchange, Alice should have item_B and Bob should have the full signature.

1) Alice starts the exchange by generating a partial signature and sending it to Bob.
2) Bob verifies the partial signature and sends item_B to Alice if the partial signature is valid.
3) Upon receiving item_B, Alice generates a full signature and sends it to Bob.
4) If Bob does not receive the full signature, he can obtain it from the arbitrator who is able to convert the partial signature to the full signature. This property guarantees that Bob will obtain the full signature if Alice refuses to send it after Step 2.

In the above scenario, Alice is usually called the "signer" and Bob is called the "verifier." The exchange between the signer and the verifier has attracted much attention from researchers on OFE, and it is also the case which OFE refers to in the remainder of this paper. (But readers should bear in mind that OFE is a broad notion and includes many other types of exchanges.)

To ensure the fairness of exchange, there are two parties who can generate full signatures in OFE: the signer and the arbitrator. Thus, full signatures can be classified (according to the signature producer) into two types: *actual signatures* (generated by the signer) and *resolved signatures* (generated by the arbitrator). It has been widely accepted that OFE protocols must have the property of "resolution ambiguity," namely valid resolved signatures should be at least computationally indistinguishable from valid actual signatures [2]–[6]. This will make the arbitrator's involvement transparent and avoid bad publicity when network failure occurs. Another motivation for resolution ambiguity is that the receiver of a "resolved signature" should not be treated by others any differently than the receiver of an "actual signature." The arbitrator in an OFE protocol with resolution ambiguity is called as a "transparent third party." However, a fully transparent third party (e.g., actual signatures and resolved signatures have identical probability distributions) could also compromise the fairness of the exchange.

In order to make the exchange fair, the arbitrator (at Step 4 in the scenario described previously) must check if the verifier has fulfilled the obligation before converting the partial signature to the full signature. Otherwise, the verifier would obtain the full signature without fulfilling the obligation and the protocol is unfair to the signer. Most OFE protocols deal with this issue in a trivial way, by assuming that the arbitrator will produce a full signature for the verifier if and only if the verifier

Manuscript received July 18, 2010; revised December 19, 2010; accepted January 11, 2011. Date of publication February 04, 2011; date of current version May 18, 2011. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Poorvi L. Vora.

X. Huang and J. Zhou are with the Institute for Infocomm Research (I^{2}R), Singapore 138632, Singapore (e-mail: xyhuang81@gmail.com; jyzhou@i2r.a-star.edu.sg).

Y. Mu, W. Susilo, and W. Wu are with the School of Computer Science and Software Engineering, University of Wollongong, Wollongong, NSW 2522, Australia (e-mail: ymu@uow.edu.au; wsusilo@uow.edu.au; ww986@uow.edu.au).

R. H. Deng is with the School of Information Systems, Singapore Management University, Singapore 178902, Singapore (e-mail: robertdeng@smu.edu.sg).

Digital Object Identifier 10.1109/TIFS.2011.2109952

can provide a valid proof of fulfilling the obligation. This assumption, however, may not hold in practice where the arbitrator could be dishonest. One of the biggest threats to OFE is the arbitrator–verifier collusion: the arbitrator converts a partial signature to a full one *without* performing any check. While it is hard to prevent such dishonest behaviors from occurring, we believe it should be at least feasible to identify the misbehaving party. In this case, a fully transparent third party is certainly not desirable.

A. Related Work

As one of the fundamental problems, OFE has been studied extensively since its introduction. As commonly known, OFE can be constructed from verifiably encrypted signature [7]–[12] or sequential two-party multisignature [13], [3]. Several OFE security properties are considered to be important, including (but not limited to): abuse-free [14], [15], accountability [8] (previously called as the verifiability of the third party in [16]), multiuser security [2], [17], [6], security in chosen-key model [4], nonrepudiation [5], [18], [19], setup-free and standalone [20], [21], signer ambiguity [22], stateless-recipient [23], impact of system failures on the fairness [24], timely termination [7], [8], and transparent third party [5] (also known as resolution ambiguity). In the following, we only review the notion of accountability, which is the focus of this paper.

Accountability: In general, accountability requires that if a desired goal of the protocol is not met then some misbehaving parties should be (rightly) blamed [25]. The introduction of accountability in OFE (and equivalently, accountable OFE) was first given in [8].

As both the signer and the arbitrator can generate full signatures, it would be desirable if a full signature contains the information of its producer, especially when misbehavior occurs: not only could the arbitrator generate full signatures without checking if the verifier has fulfilled the obligation, the signer and the verifier could also frame the third party who does not misbehave [8]. The purpose of accountable OFE is to identify the party who is responsible for the full signature, and thus force the arbitrator and the signer to behave honestly when generating full signatures. This requires that actual signatures (generated by the signer) be distinguishable from resolved signatures (generated by the arbitrator).

As shown in [8], an OFE protocol will be accountable if actual signatures and resolved signatures have obviously different forms: given a valid full signature, one (with only public information) can easily tell who is the full signature generator. To the best of our knowledge, it is the only known design of accountable OFE. However, the solution violates the requirement of transparent third party, where resolved signatures should be at least computationally indistinguishable from actual signatures. This partly contributes to explaining the lack of further research on accountable OFE since its introduction a decade ago.

B. Motivations and Contributions

This paper is motivated by the following two questions:

1) What are the properties an accountable OFE protocol should possess?
2) How can we design an accountable OFE protocol with a transparent third party?

As an important notion to solve practical issues, accountable OFE received much less attention than what it deserves. To date, it remains unknown what exactly an accountable OFE protocol is and what properties it should have, although the notion has a ten-year history. The lack of satisfactory answers to those questions has hindered the development of the research on accountable OFE. We believe it is worthwhile (both in theory and in practice) to formally investigate accountable OFE and provide its precise definitions. On the other hand, accountability and transparency (equivalently, transparent third party) seem to contradict each other [26] and cannot coexist in OFE. Accountability requires that actual signatures and resolved signatures be distinguishable, while transparency requires them be (at least computationally) indistinguishable. To date, there are no OFE protocols accommodating both properties. (The existing design of accountable OFE [8] has a nontransparent third party.)

The contributions of this paper lie in the following aspects:

1) The definition of OFE is refined by introducing three new algorithms, which we believe can capture the essential requirements of accountable OFE.
2) We introduce several security notions an accountable OFE protocol should satisfy. Each notion captures a potential attack on the accountability of OFE and is defined in a formal game-based model.
3) We provide a generic (also the first) design of OFE with transparency and accountability, where the arbitrator's involvement is (computationally) transparent and the accountability does not rely on any other third parties. The identity of the full signature producer remains unknown until certain proofs are issued, either by the signer or by the arbitrator.
4) The generic protocol is based on well-studied cryptographic primitives. We prove (in the random oracle model) that it satisfies all security requirements defined in this paper, assuming that the underlying primitives satisfy certain security notions. In addition, we provide an efficient instantiation and two variants of the generic protocol.

Section II reviews the existing definition of normal OFE and defines accountable OFE.

II. Formal Definitions of Accountable OFE

This section formally defines accountable OFE and its security notions. We first review the definition of a normal OFE protocol [2], [27].

Definition 1: An OFE protocol is made up of seven algorithms: $\mathsf{Setup}^{\mathsf{TTP}}$, $\mathsf{Setup}^{\mathsf{User}}$, PSig, PVer, Sig, Ver, and Res. Given a security parameter Param, these algorithms are defined as follows.

1) $\mathsf{Setup}^{\mathsf{TTP}}(\mathsf{Param}) \rightarrow (\mathsf{APK}, \mathsf{ASK})$. On input the parameter Param, the arbitrator executes this algorithm to obtain a public–private key pair $(\mathsf{APK}, \mathsf{ASK})$.
2) $\mathsf{Setup}^{\mathsf{User}}(\mathsf{Param}) \rightarrow (\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i})$. On input the parameter Param, the signer U_i executes this algorithm to obtain a public–private key pair $(\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i})$.

3) $\mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}) \rightarrow \sigma^P$. On input a message m, U_i's private key SK_{U_i} and the arbitrator's public key APK, the partial signing algorithm PSig outputs a partial signature σ^P.
   Here, m should include the description of the items to be exchanged between U_i and U_j. This will help the arbitrator process the resolution request. Notice that the partial signature σ^P is not U_i's agreement of sending its item to U_j, even though in some concrete constructions σ^P is a classical digital signature. Instead, only a full signature (will be defined shortly) can be viewed as U_i's agreement of fulfilling its obligation (as described in m). The partial signature σ^P only shows U_i's willingness to send its item to U_j if U_j fulfills its obligation.
4) $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK}) \rightarrow \{0, 1\}$. On input a pair (m, σ^P) and two public keys $(\mathsf{PK}_{U_i}, \mathsf{APK})$, this algorithm outputs "1" (if σ^P is valid) or "0" (otherwise). (m, σ^P) is said to be a valid message–partial-signature pair under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ if $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$.
5) $\mathsf{Sig}(m, \mathsf{SK}_{U_i}, \sigma^P, \mathsf{APK}) \rightarrow \sigma$. On input a message m, U_i's private key SK_{U_i}, (optionally) U_i's valid partial signature σ^P on m and (optionally) the arbitrator's public key APK, this algorithm outputs a full signature σ.
   To distinguish from the partial signature σ^P generated by PSig, σ is called the full signature.
6) $\mathsf{Ver}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}) \rightarrow \{0, 1\}$. On input a pair (m, σ) and two public keys $(\mathsf{PK}_{U_i}, \mathsf{APK})$, this algorithm outputs "1" (if σ is valid) or "0" (otherwise).
   (m, σ) is said to be a valid message–signature pair under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ if $\mathsf{Ver}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$.
7) $\mathsf{Res}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{ASK}) \rightarrow \sigma$. As a resolution algorithm, it is executed by the arbitrator if the verifier U_j does not receive the full signature from U_i, but has the signer U_i's valid partial signature σ^P on m and has fulfilled the obligation to U_i. On input a valid message–partial-signature pair (m, σ^P) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and the arbitrator's private key ASK, this algorithm outputs a full signature σ.

Correctness: The verification algorithms PVer and Ver will output "1" if signatures are generated according to the protocol specification:

1) $\mathsf{PVer}(m, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$;
2) $\mathsf{Ver}(m, \mathsf{Sig}(m, \mathsf{SK}_{U_i}, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{APK}), \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$; and
3) $\mathsf{Ver}(m, \mathsf{Res}(m, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{PK}_{U_i}, \mathsf{ASK}), \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$.

Remark: Definition 1 is essentially the same as those in [2] and [27], but with the only difference that the partial signature σ^P is allowed as an optional input to Sig. This is due to the fact that σ^P does exist when U_i runs the algorithm Sig.

Security in the Multiuser Setting: Due to [2], [6], an OFE protocol must satisfy the following security requirements in the multiuser setting.

1) *Security against Signers*: The signer should not be able to produce a valid partial signature which cannot be transformed into a valid full signature by an honest arbitrator.
2) *Security against Verifiers*: The verifier should not be able to transform a valid partial signature into a valid full signature, without explicitly asking the arbitrator to do so.
3) *Security against the Arbitrator*: The arbitrator should not be able to produce a valid full signature on message m without explicitly asking the signer to produce a partial signature on m.

Due to page limitation, these notions will not be revisited in this paper. Please refer to [2], [6] for their formal definitions.

A. Accountable OFE

As we have briefly introduced in Section I-A, a valid full signature of an accountable OFE protocol must provide the information about the signature producer, such that it will be feasible to verify whether a full signature is generated by the signer or by the arbitrator. At the same time, we want to design an accountable OFE protocol where the arbitrator's involvement in the exchange is transparent. A promising approach to achieve our purpose is to keep the identity of the full signature producer unknown to outsiders until a piece of proof is issued, namely:

1) Given a valid full signature and other public information, it is computationally infeasible to tell who is the signature producer.
2) Given a valid full signature, both the signer and the arbitrator can independently generate evidence of the signature producer's identity. This will identify the entity who, either the signer or the arbitrator, should take the full responsibility for generating the full signature.

With this in mind, we first introduce the algorithms of which an accountable OFE protocol consists.

Definition 2: An accountable OFE protocol is made up of ten algorithms: $\mathsf{Setup}^{\mathsf{TTP}}$, $\mathsf{Setup}^{\mathsf{User}}$, PSig, PVer, Sig, Ver, Res, $\mathsf{Prove}^{\mathsf{User}}$, $\mathsf{Prove}^{\mathsf{TTP}}$, and Open. With a parameter Param, the first seven algorithms are the same as those in Definition 1 and the remaining three are defined as follows:

8) $\mathsf{Prove}^{\mathsf{User}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{SK}_{U_i}) \rightarrow \{\pi^{\mathsf{User}}_{\mathsf{Claim}}, \pi^{\mathsf{User}}_{\mathsf{Deny}}\}$.
   On input a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and the signer's private key SK_{U_i}, this algorithm outputs a proof $\pi^{\mathsf{User}}_{\mathsf{Claim}}$ to claim the signature (if σ is an output of the algorithm Sig) or a proof $\pi^{\mathsf{User}}_{\mathsf{Deny}}$ to deny the signature (in all other cases).
9) $\mathsf{Prove}^{\mathsf{TTP}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{ASK}) \rightarrow \{\pi^{\mathsf{TTP}}_{\mathsf{Claim}}, \pi^{\mathsf{TTP}}_{\mathsf{Deny}}\}$.
   On input a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and the arbitrator's private key ASK, this algorithm outputs a proof $\pi^{\mathsf{TTP}}_{\mathsf{Claim}}$ to claim the signature (if σ is an output of the algorithm Res) or a proof $\pi^{\mathsf{TTP}}_{\mathsf{Deny}}$ to deny the signature (in all other cases).
10) $\mathsf{Open}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \pi) \rightarrow \{\mathsf{PK}_{U_i}, \mathsf{APK}, \perp\}$. On input a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$, and a proof π, the output of this algorithm is "PK_{U_i}" (if π can prove σ as an output of the algorithm Sig), "APK" (if π can prove σ as an output of the algorithm Res), or the symbol "$\perp$" (if π does not contain enough information to identify the full signature producer, e.g., π is a random string).

Correctness: In addition to those required in normal OFE, there is an additional requirement in accountable OFE: Given a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and a proof π generated by the algorithm $\mathsf{Prove}^{\mathsf{User}}$ or $\mathsf{Prove}^{\mathsf{TTP}}$,

$\mathsf{Open}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \pi) \neq \perp$. In other words, any valid full signatures can be either claimed or denied.

Three remarks on Definition 2.

1) The proof-generation needs the private key SK_{U_i} or ASK. This allows the signer and the arbitrator to independently generate the proof.
2) The private key SK_{U_i} or ASK is the only secret information needed to generate the proof. There is no need to record other secret information (e.g., random numbers) that one used during the generation of full signatures.
3) The Open algorithm does not need the assistance from any additional third parties, since those parties could refuse to provide the assistance when colluding with the signer or the arbitrator. The proof generated by $\mathsf{Prove}^{\mathsf{TTP}}$ or $\mathsf{Prove}^{\mathsf{TTP}}$, together with other public information, is sufficient to identify the full signature producer.

B. *Oracles Accessible by Adversaries*

We now define the security notions an accountable OFE protocol should satisfy. Accountable OFE must satisfy the security requirements reviewed in Definition 1, which are essential for all OFE protocols. The focus of this section is to define the accountability in OFE. As we shall show, it consists of several security notions. Each security notion is motivated by a potential attack on the accountability of OFE, followed by a game-based model between the adversary and the challenger. In most games, the challenger will generate the arbitrator's key pair $(\mathsf{APK}, \mathsf{ASK})$ and send the public key APK to the adversary, who can make queries to the following oracles.

- $\mathcal{O}^{\mathsf{UserCreate}}$. This oracle executes the algorithm $\mathsf{Setup}^{\mathsf{User}}$ and generates a key pair $(\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i})$ for the user U_i. It also maintains a list $\mathcal{L}^{\mathsf{PK}}$ which is initially empty. After creating the user U_i, this oracle adds $(\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i})$ on the list $\mathcal{L}^{\mathsf{PK}}$ and sends PK_{U_i} to the adversary. In this case, we say U_i has been created. Notice that other oracles only answer queries on created users.
- $\mathcal{O}^{\mathsf{Corruption}}$. On input the public key PK_{U_i} of a created user, this oracle outputs its private key SK_{U_i}.
- $\mathcal{O}^{\mathsf{PSig}}$. On input a pair (m, PK_{U_i}), this oracle runs $\mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK})$ and responds with a partial signature.
- $\mathcal{O}^{\mathsf{Sig}}$. On input a triple $(m, \sigma^P, \mathsf{PK}_{U_i})$ satisfying $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$, this oracle runs $\mathsf{Sig}(m, \sigma^P, \mathsf{SK}_{U_i}, \mathsf{APK})$ and responds with an actual signature.
- $\mathcal{O}^{\mathsf{Res}}$. On input a triple $(m, \sigma^P, \mathsf{PK}_{U_i})$ satisfying $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$, this oracle runs $\mathsf{Res}(m, \sigma^P, \mathsf{ASK}, \mathsf{PK}_{U_i})$ and responds with a resolved signature.
- $\mathcal{O}^{\mathsf{Prove}}$. On input a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and a symbol $id \in \{\mathsf{User}, \mathsf{TTP}\}$, this oracle responds with a proof by running $\mathsf{Prove}^{\mathsf{User}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{SK}_{U_i})$ if $id = \mathsf{User}$, or $\mathsf{Prove}^{\mathsf{TTP}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{ASK})$ if $id = \mathsf{TTP}$.

In the random oracle model, adversaries are allowed to make requests to random oracles. We will use the following two notations in the security definition.

- (t, q)-adversary: A probabilistic adversary who can make up to q queries in polynomial time t.
- $Q \nrightarrow \{\mathcal{O}^1, \mathcal{O}^2, \ldots, \mathcal{O}^n\}$: The query Q cannot appear as a request to any oracle $\mathcal{O} \in \{\mathcal{O}^1, \mathcal{O}^2, \ldots, \mathcal{O}^n\}$.

C. *Accountability*

We define three security notions which we believe are necessary to make an OFE protocol accountable.

Type I Accountability: This notion is motivated by the following attack: a dishonest signer wishes to generate a full signature which can be proved as an output of the algorithm Res. Such a successful attack will enable the signer to frame the arbitrator for generating a full signature (which is actually generated by the signer). The attack is formally defined as follows.

Phase 1: The challenger generates the arbitrator's key pair $(\mathsf{APK}, \mathsf{ASK})$ by running $\mathsf{Setup}^{\mathsf{TTP}}$. The adversary is given the arbitrator's public key APK.

Phase 2: The adversary can adaptively make queries to all oracles in Section II-B. At the end of this phase, the adversary generates a public key PK^* and the challenger adds the pair $(\mathsf{PK}^*, \top)$ to $\mathcal{L}^{\mathsf{PK}}$, which means that PK^* has been created but the challenger may not know the corresponding private key.

Phase 3: The adversary continues making queries to $\mathcal{O}^{\mathsf{Res}}$ and $\mathcal{O}^{\mathsf{Prove}}$ with the restriction that $(\cdot, \cdot, \mathsf{PK}^*, \mathsf{User}) \nrightarrow \mathcal{O}^{\mathsf{Prove}}$. We disallow such queries since PK^* is provided by the adversary and the challenger may not have the private key.

Phase 4: The adversary outputs a valid message–signature pair $(\hat{m}, \hat{\sigma})$ under $(\mathsf{PK}^*, \mathsf{APK})$ and a proof $\hat{\pi}$ such that $(\hat{m}, \cdot, \mathsf{PK}^*) \nrightarrow \mathcal{O}^{\mathsf{Res}}$.

The success probability that the adversary has in the game is $\Pr[\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}) = \mathsf{APK}]$. An OFE protocol is (t, q, ϵ)-Type-I-Accountable if no (t, q)-adversary can have success probability more than ϵ in the above game.

Type II Accountability: The attacking scenario associated with Type II accountability is to simulate a dishonest arbitrator, who wishes to generate a full signature that can be proved as an output of the algorithm Sig. As the counterpart of the attack considered in Type I accountability, such a successful attack will make the protocol unfair to the signer. Type II accountability is formally defined as follows.

Phase 1: The adversary makes a query to $\mathcal{O}^{\mathsf{UserCreate}}$ and lets the response be a public key PK^*. After that, the adversary generates the arbitrator's public key APK and sends it to the challenger.

Phase 2: The adversary can make queries to all oracles in Section II-B except $\mathcal{O}^{\mathsf{Res}}$, with the restriction that $(\cdot, \cdot, \cdot, \mathsf{TTP}) \nrightarrow \mathcal{O}^{\mathsf{Prove}}$. (Recall that APK is generated by the adversary.)

Phase 3: The adversary outputs a valid pair $(\hat{m}, \hat{\sigma})$ under $(\mathsf{PK}^*, \mathsf{APK})$ and a proof $\hat{\pi}$ such that $(\hat{m}, \cdot, \mathsf{PK}^*) \nrightarrow \mathcal{O}^{\mathsf{Sig}}$ and $\mathsf{PK}^* \nrightarrow \mathcal{O}^{\mathsf{Corruption}}$.

The success probability that the adversary has in the game is $\Pr[\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}) = \mathsf{PK}^*]$. An OFE protocol is (t, q, ϵ)-Type-II-Accountable if no (t, q)-adversary can have success probability more than ϵ in the above game.

Type III Accountability: Another threat on accountability is that the signer and the arbitrator can both claim (or, deny) a valid full signature σ. As an example, the evidence generated by the signer proves that the arbitrator is the producer of σ, while the arbitrator proves that σ is generated by the signer. We believe an OFE protocol should be accountable; even the signer and the arbitrator collude with each other by sharing all secret information, and generating the signature and the proof in a cooperative way. This is formally defined as follows.

Phase 1: Given the system parameter, the adversary outputs two different public keys PK^* and APK.

Since both keys are generated by the adversary, there is no need to provide the adversary with any oracle access.[1]

Phase 2: The adversary outputs a valid message–signature pair $(\hat{m}, \hat{\sigma})$ under $(\mathsf{PK}^*, \mathsf{APK})$ and two proofs $(\hat{\pi}^{\mathsf{User}}, \hat{\pi}^{\mathsf{TTP}})$. The adversary wins the game if one of the following two requirements is satisfied:

1) $\hat{\sigma}$ can be both claimed by the signer and the arbitrator, i.e., $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}^{\mathsf{User}}) = \mathsf{PK}^*$ and $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}^{\mathsf{TTP}}) = \mathsf{APK}$; or
2) $\hat{\sigma}$ can be both denied by the signer and the arbitrator, i.e., $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}^{\mathsf{User}}) = \mathsf{APK}$ and $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}^{\mathsf{TTP}}) = \mathsf{PK}^*$.

An OFE protocol is (t, ϵ)-Type-III-Accountable if no t-time adversary can win the game with probability more than ϵ.

Remark: An OFE protocol with Type III accountability does not guarantee Type I accountability or Type II accountability. As an example, an adversary can break Type I accountability by successfully forging an output of Res but cannot prove it as an output of Sig.

Definition 3 (Accountable OFE): An accountable OFE protocol must be Type I Accountable, Type II Accountable, and Type III Accountable.

D. Transparent Third Party

This section is devoted to the definition of transparent third party. In an OFE protocol, transparent third party (equivalently, resolution ambiguity) requires that it should be (at least computationally) infeasible to distinguish a valid actual signature (generated by the signer) from a valid resolved signature (generated by the arbitrator). It can protect the privacy of a full signature generator and avoid bad publicity. This property is defined by the following game.

Phase 1: The challenger generates the arbitrator's key pair $(\mathsf{APK}, \mathsf{ASK})$ by running the algorithm $\mathsf{Setup}^{\mathsf{TTP}}$. The adversary is given APK.

Phase 2: The adversary can adaptively issue queries to all oracles in Section II-B.

At the end of this phase, the adversary outputs a message $\hat{m}$, a partial signature $\hat{\sigma}^P$, and a public key PK^* (which is generated by $\mathcal{O}^{\mathsf{UserCreate}}$), with restrictions that $\mathsf{PK}^* \nrightarrow \mathcal{O}^{\mathsf{Corruption}}$ and $\mathsf{PVer}(\hat{m}, \hat{\sigma}^P, \mathsf{PK}^*, \mathsf{APK}) = 1$.

Phase 3: Upon receiving $(\hat{m}, \hat{\sigma}^P, \mathsf{PK}^*)$, the challenger picks a random bit $b \in \{0, 1\}$ and generates the full signature $\hat{\sigma}$ accordingly. If $b = 0$, $\hat{\sigma} = \mathsf{Sig}(\hat{m}, \mathsf{SK}^*, \hat{\sigma}^P, \mathsf{APK})$. Otherwise, $\hat{\sigma} = \mathsf{Res}(\hat{m}, \hat{\sigma}^P, \mathsf{PK}^*, \mathsf{ASK})$. In either case, $\hat{\sigma}$ is given to the adversary.

[1]An exception is that if the security of the protocol is considered in the random oracle model, the adversary can make requests to random oracles before generating public keys.

Phase 4: Given $\hat{\sigma}$, the adversary can continue making requests to all oracles, but with restrictions that $\mathsf{PK}^* \nrightarrow \mathcal{O}^{\mathsf{Corruption}}$ and $(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \cdot) \nrightarrow \mathcal{O}^{\mathsf{Prove}}$. At the end of this phase, the adversary outputs its guess $b' \in \{0, 1\}$.

The success probability that the adversary has in the game is $\Pr[b = b']$.

Definition 4 (Transparent Third Party): The third party in an OFE protocol is (t, q, ϵ)-Transparent if no (t, q)-adversary can have success probability more than $\epsilon + 1/2$ in the above game.

Remark: Definition 4 is essentially the same as the property "ambiguity" defined in [2], which requires that valid actual signatures be computationally indistinguishable from valid resolved signatures.

Next section briefly introduces the preliminaries needed for our design of accountable OFE with a transparent third party.

III. Preliminaries

Before presenting the details, we briefly review some cryptographic primitives which will be used in our construction.

A. (Strongly) Existentially Unforgeable Digital Signatures

A signature scheme consists of three algorithms (*KeyGen*, *Sign*, *Ver*) associated with key generation, signing, and verification, respectively. The signer runs *KeyGen* to generate a private–public key pair (sk, pk). On input a private key sk and a message m, the *Sign* algorithm generates a signature σ on m. On input a message–signature pair (m, σ) and a public key pk, the verification algorithm *Ver* outputs "1" if σ is a valid signature, or "0" otherwise. We review the security requirements of digital signatures as follows.

The probabilistic polynomial-time (PPT) adversary is allowed to adaptively make queries to a signing oracle, namely the adversary can query the signing oracle to obtain valid signatures of adaptively chosen messages. Let $M = \{m_i\}$ be the set of messages chosen by the adversary and $S = \{(m_i, \sigma_i)\}$ be the set of corresponding message–signature pairs. After all queries are made, the adversary outputs a pair (m^*, σ^*) and wins the game if $Ver(m^*, \sigma^*, pk) = 1$. We say a signature scheme is $(t, q, \epsilon_{\mathrm{EU}})$—*Existentially Unforgeable*—f no (t, q)-adversary can win the game with probability more than ϵ_{EU} and $m^* \notin M$. We say a signature scheme is $(t, q, \epsilon_{\mathrm{SEU}})$—*Strongly Existentially Unforgeable*—f no (t, q)-adversary can win the game with probability more than ϵ_{SEU} and $(m^*, \sigma^*) \notin S$. Several generic methods [28], [27], [29], [30] have been proposed to convert an existentially unforgeable signature scheme to a strongly existentially unforgeable signature scheme.

B. SPK: Signatures Based on Proofs of Knowledge

Let $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ be a binary relation. We say that $\mathcal{R}$ is polynomially bounded if there exists a polynomial p such that $|y| \le p(|x|)$ for any $(x, y) \in \mathcal{R}$. We say that $\mathcal{R}$ is an **NP**-relation if it is polynomially bounded and there exists a polynomial-time algorithm for deciding membership in $\mathcal{R}$. The **NP**-language $L_{\mathcal{R}}$ associated with R is the set of x for which there exists y such that $(x, y) \in \mathcal{R}$. A proof of knowledge [31] for an **NP**-relation $\mathcal{R}$ is a protocol whereby there is a common input x to the prover and the verifier, and a private input y to the prover. The prover tries to convince the verifier that $(x, y) \in \mathcal{R}$. One example of the proof of knowledge is the Σ-protocol with correctness, special soundness, and special (honest-verifier) zero-knowledge [32] (e.g., GQ protocol [33] and Schnorr protocol [34]). It is known that any language in **NP** has a Σ-protocol if one-way functions exist [35], [36]. Given two Σ-protocols, i.e., Σ_1 for $\mathcal{R}_1$ and Σ_2 for $\mathcal{R}_2$, we can construct another Σ-protocol Σ_{OR} (called OR-proof) [32] which allows the prover to prove that given two inputs x_1, x_2, he knows y such that either $(x_1, y) \in \mathcal{R}_1$ or $(x_2, y) \in \mathcal{R}_2$ without revealing which is the case (called the witness indistinguishability property [37]).

One can apply the Fiat–Shamir heuristic [38] to convert a Σ-protocol into a signature on a message m. This is called as signatures based on proofs of knowledge in [39] (or, SPK for short) and denoted by $\mathrm{SPK}\{\mathsf{SK} : \mathsf{Statement}\}(m)$. An SPK shows that a person who knows a witness SK such that $(\mathsf{SK}, \mathsf{Statement})$ in a relation $\mathcal{R}$ has signed the message m. In particular, $\mathsf{Statement}$ can be the combination of two or more statements with the form "$\mathsf{ST}_1 \wedge \mathsf{ST}_2 \wedge \cdots$" (all statements are true) or "$\mathsf{ST}_1 \vee \mathsf{ST}_2 \vee \cdots$" (one of the statements is true).

As a signature scheme, the existential unforgeability of SPK can be proved in the random oracle model [40], [41]. An SPK is said to be *complete* if the $\mathsf{Statement}$ is true and the prover with the knowledge of SK can always be successful to generate a proof to convince the verifier. An SPK is (t, ϵ_S)-*Sound* if the $\mathsf{Statement}$ is false, then no t-time prover can convince the verifier that the $\mathsf{Statement}$ is true with success probability more than ϵ_S. An SPK is (t, ϵ_V)-*Valid* if an algorithm $\mathcal{A}$ can successfully convince the verifier with probability $\varepsilon > \epsilon_V$, then there is a t-time knowledge extractor who can use $\mathcal{A}$ to extract the secret SK with probability more than $\varepsilon - \epsilon_V$. Detailed definitions of those notions can be found in [31] and [39].

C. Undeniable Signatures

The concept of undeniable signatures was introduced by Chaum and van Antwerpen [42]. The most distinctive feature of undeniable signatures is that signature verification needs the assistance from the signer: An undeniable signature can only be verified by someone who knows the private key, or by a proof generated by the private key owner.

An undeniable signature scheme $\mathsf{US}\text{-}\Sigma$ is made up of following algorithms:

- $\mathsf{US.KeyGen}$. On input the parameter $\mathsf{US.Param}$ (including a security number κ and the message space $\mathcal{M}$), the user runs this algorithm to generate a key pair $(\mathsf{PK}, \mathsf{SK}) \in \mathcal{R}_{\mathsf{Key}}$. Here, $\mathcal{R}_{\mathsf{Key}}$ is an **NP**-relation specified in $\mathsf{US.Param}$.
- $\mathsf{US.Sign}/\mathsf{US.Verify}$. Given a message $m \in \mathcal{M}$, one can generate an undeniable signature σ_{US} using SK and the signing algorithm $\mathsf{US.Sign}$. The verification algorithm $\mathsf{US.Verify}$, by taking the input $(m, \sigma_{\mathsf{US}})$ and $(\mathsf{PK}, \mathsf{SK})$, outputs "1" (if σ_{US} is valid) or "0" (otherwise). Let $\mathcal{R}_{\mathsf{PK}} = \{(m, \sigma_{\mathsf{US}}) : \mathsf{US.Verify}(m, \sigma_{\mathsf{US}}, \mathsf{SK}, \mathsf{PK}) = 1\}$.
- $\mathsf{Confirmation}/\mathsf{Disavowal}$. Given a message–signature pair $(m, \sigma_{\mathsf{US}})$ and a public key PK, the prover uses the corresponding private key SK to prove $(m, \sigma_{\mathsf{US}}) \in \mathcal{R}_{\mathsf{PK}}$ by running the $\mathsf{Confirmation}$ protocol with the verifier. Otherwise, the prover uses SK to prove $(m, \sigma_{\mathsf{US}}) \notin \mathcal{R}_{\mathsf{PK}}$ by running the $\mathsf{Disavowal}$ protocol. In undeniable signatures, these two protocols are usually full-fledge zero-knowledge protocols.

  To serve the purpose of this paper, we require that one can use the private key SK to generate efficient SPKs for $(m, \sigma_{\mathsf{US}}) \in \mathcal{R}_{\mathsf{PK}}$ and $(m, \sigma_{\mathsf{US}}) \notin \mathcal{R}_{\mathsf{PK}}$. Such SPKs exist in several undeniable signature schemes, especially those based on the discrete logarithm problem and its variants [42], [43]. A concrete example is given at the end of this section. (Note that those SPKs cannot be used as confirmation or disavowal protocols since they are ordinary signatures.)

The correctness requires that $\mathsf{US.Verify}(m, \mathsf{US.Sign}(m, \mathsf{SK}, \mathsf{PK}), \mathsf{SK}, \mathsf{PK}) = 1$.

We list the security notions an undeniable signature scheme should satisfy. In all definitions, the adversary is allowed to make (in an adaptive manner) $\mathsf{US.Sign}$ and $\mathsf{Confirmation}/\mathsf{Disavowal}$ queries to the challenger, who responds with the output of the corresponding algorithm.

Unforgeability: Given a public key PK, the adversary outputs a pair $(m^*, \sigma^*_{\mathsf{US}})$ after all queries are made, with the restriction that m^* is a fresh message, i.e., m^* does not appear as one of $\mathsf{US.Sign}$ queries. An undeniable signature scheme is $(t, q, \epsilon^{\mathrm{EU}}_{\mathsf{US}})$-Existentially Unforgeable if no (t, q)-adversary can create a pair $(m^*, \sigma^*_{\mathsf{US}}) \in \mathcal{R}_{\mathsf{PK}}$ with probability more than $\epsilon^{\mathrm{EU}}_{\mathsf{US}}$.

Anonymity: Roughly speaking, this notion requires that the adversary be unable to tell who generates an undeniable signature. Given two public keys PK_0 and PK_1, the adversary can issue queries adaptively. In a certain phase, the adversary sends a fresh message m^* to the challenger, who flips a random coin $b \in \{0, 1\}$ and generates a signature σ^*_{US} such that $(m^*, \sigma^*_{\mathsf{US}}) \in \mathcal{R}_{\mathsf{PK}_b}$. After receiving σ^*_{US}, the adversary can continue making queries with restrictions that m^* cannot be chosen as one of $\mathsf{US.Sign}$ queries or $\mathsf{Confirmation}/\mathsf{Disavowal}$ queries. After all queries are made, the adversary outputs a bit b'. Let $\Pr[b = b']$ be the success probability that the adversary has in the game. We say an undeniable signature scheme is $(t, q, \epsilon^{A}_{\mathsf{US}})$-Anonymous if no (t, q)-adversary can have success probability more than $\epsilon^{A}_{\mathsf{US}} + 1/2$.

Notice that some undeniable signature schemes (e.g., [44]) have a stronger anonymity than we defined above, in the sense that the adversary is allowed to make queries adaptively with the only restriction that $(m^*, \sigma^*_{\mathsf{US}})$ cannot appear as one of $\mathsf{Confirmation}/\mathsf{Disavowal}$ queries. Such schemes certainly satisfy the above definition of anonymity.

Collision-Resistance in Undeniable Signatures: In the game of collision-resistance in undeniable signatures, the adversary is allowed to generate two different public keys PK_0 and PK_1. The adversary wins the game if it can output a pair $(m, \sigma_{\mathsf{US}})$ such that $(m, \sigma_{\mathsf{US}}) \in \mathcal{R}_{\mathsf{PK}_0} \cap \mathcal{R}_{\mathsf{PK}_1}$, i.e., σ_{US} is a valid undeniable signature under both PK_0 and PK_1. We say an undeniable signature scheme is $(t, \epsilon^{\mathrm{CR}}_{\mathsf{US}})$-Collision-Resistant if no t-time adversary can win the above game with probability more than $\epsilon^{\mathrm{CR}}_{\mathsf{US}}$. Below is a concrete undeniable signature scheme with collision-resistance.

Below is a concrete scheme which satisfies all the above security requirements.

The FDH Variant of Chaum's Undeniable Signature [42]: The $\mathsf{US.Param}$ is $(G, p, \kappa, g, H_{\mathsf{US}})$, where G is an Abelian group of prime order $p \geq 2^{\kappa}$, g is a generator of G, and H_{US} is a hash function: $\{0,1\}^* \to G$.

- $\mathsf{US.KeyGen}$. This algorithm outputs a key pair $(\mathsf{SK}, \mathsf{PK})$, where SK is a random element in $\mathbb{Z}_p$ and $\mathsf{PK} = g^{\mathsf{SK}}$. The **NP**-relation $\mathcal{R}_{\mathsf{Key}}$ is defined as $\{(\mathsf{SK}, \mathsf{PK}) : \mathsf{SK} \in \mathbb{Z}_p \text{ and } \mathsf{PK} = g^{\mathsf{SK}}\}$.
- $\mathsf{US.Sign}$ and $\mathsf{US.Verify}$. For a message m, the undeniable signature is $\sigma_{\mathsf{US}} = H_{\mathsf{US}}(m)^{\mathsf{SK}}$. To verify an undeniable signature σ_{US}, algorithm $\mathsf{US.Verify}$ will output "1" if $\sigma_{\mathsf{US}} = H_{\mathsf{US}}(m)^{\mathsf{SK}}$ and "0" otherwise.
- $\mathsf{Confirmation/Disavowal}$. The available protocols are recently revisited in [45], whose details are omitted here.

  One can generate efficient SPKs to prove whether a given undeniable signature is valid or not, by applying Fiat–Shamir heuristic [38] to the three-move honest-verifier zero-knowledge proof for Chaum's scheme. The protocol is given in [46] and revisited in [45].

The FDH variant of Chaum's scheme with NIZK confirmation and disavowal protocols is existentially unforgeable if the computational Diffie–Hellman problem is infeasible in G [45]. The anonymity of Chaum's scheme can be reduced to the hardness of the decisional Diffie–Hellman problem in G [47]. Chaum's undeniable signature is also collision-resistant: If σ_{US} is a valid undeniable signature under two public keys PK_1 and PK_2, then $\mathsf{PK}_1 = \mathsf{PK}_2$. This contradicts the requirement that PK_1 be different from PK_2.

D. Collision-Resistant Hashing

Let $\mathcal{H} = \{H_k\}$ be a keyed hash family of functions $H_k : \mathsf{Input}_k \to \mathsf{Output}_k$ indexed by $k \in \mathcal{K}$. We say that algorithm $\mathcal{A}$ has advantage ϵ_H in breaking the collision-resistance of $\mathcal{H}$ if $\Pr[\mathcal{A}(k) = (m_0, m_1) : m_0 \neq m_1, H_k(m_0) = H_k(m_1)] \geq \epsilon_H$, where the probability is over the random choice of k and the random bits of $\mathcal{A}$. We say a hash family $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant if no t-time adversary can have advantage more than ϵ_H in breaking the collision-resistance of $\mathcal{H}$.

IV. Generic Design of Accountable OFE With A Transparent Third Party

This section is devoted to the description and the analysis of a generic design of accountable OFE with a transparent third party.

We start from the generic construction of multiuser secure OFE proposed in [2]. The partial signature in [2] is an ordinary signature of the signer, and the full signature consists of a partial signature and an OR-Signature, which can be generated by the signer or by the arbitrator. Our construction employs a similar idea but has the following differences.

1) The signer and the arbitrator generate their key pairs in an undeniable signature scheme.
2) The partial signature is a signature based on proof of knowledge (SPK) generated by the signer using his/her private key.
3) The full signature consists of four components $(\rho, r_\rho, \chi, \eta)$: (1) ρ is the partial signature, (2) r_ρ is a random salt, (3) χ is an undeniable signature, and (4) η is another SPK proving that χ is a valid undeniable signature generated by the signer or by the arbitrator.

This completes the high-level description of our design, whose details are given as follows.

A. Generic Construction

Let $\mathsf{US.Param}$ be the parameter in an undeniable signature scheme $\mathsf{US}\text{-}\Sigma = (\mathsf{US.KeyGen}, \mathsf{US.Sign}, \mathsf{US.Verify}, \mathsf{Confirmation}, \mathsf{Disavowal})$. We need three hash functions $H_1 : \{0,1\}^* \to \mathcal{M}$ and $H_2, H_3 : \{0,1\}^* \to \mathbb{Z}_{2^\kappa}$, which are chosen from the collision-resistant hash family $\mathcal{H}$. Here, $\mathcal{M}$ is the message space and κ is the security number defined in $\mathsf{US.Param}$. The parameter in our construction is $\mathsf{Param} = \{\mathsf{US.Param}, \mathsf{US}\text{–}\Sigma, H_1, H_2, H_3\}$.

1) $\mathsf{Setup}^{\mathsf{TTP}}(\mathsf{Param})$. Given Param, the arbitrator runs the algorithm $\mathsf{US.KeyGen}$ to obtain a pair $(\mathsf{APK}, \mathsf{ASK}) \in \mathcal{R}_{\mathsf{Key}}$. Here, $\mathcal{R}_{\mathsf{Key}}$ is an **NP**-relation defined in $\mathsf{US.Param}$.
2) $\mathsf{Setup}^{\mathsf{User}}(\mathsf{Param})$. Given Param, the signer U_i runs $\mathsf{US.KeyGen}$ to obtain a pair $(\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i}) \in \mathcal{R}_{\mathsf{Key}}$.
3) $\mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK})$. Given a message m, U_i uses SK_{U_i} to calculate $\rho = \mathrm{SPK}\{\mathsf{SK}_{U_i} : (\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i}) \in \mathcal{R}_{\mathsf{Key}}\}(m)$. The partial signature σ^P of m is ρ.
4) $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK})$. Given m and $\sigma^P = \rho$, one can verify if ρ is a correct $\mathrm{SPK}\{\mathsf{SK}_{U_i} : (\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i}) \in \mathcal{R}_{\mathsf{Key}}\}(m)$. This algorithm outputs "1" if it is correct. Otherwise, it outputs "0."
5) $\mathsf{Sig}(m, \mathsf{SK}_{U_i}, \sigma^P, \mathsf{APK})$. Given a message m, its valid partial signature $\sigma^P = \rho$ and the arbitrator's public key APK, U_i will choose a random element $r_\rho \in \mathbb{Z}_{2^\kappa}$ and calculate (χ, η) as follows:
   - U_i uses the private key SK_{U_i} to generate an undeniable signature χ on the message $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$. That is, $\chi = \mathsf{US.Sign}(\mathfrak{m}_1, \mathsf{SK}_{U_i})$ and $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{PK}_{U_i}}$.
   - U_i then proves that χ is a valid undeniable signature under PK_{U_i} or APK, by calculating (an OR-proof) $\eta = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_1 \vee \mathsf{ST}_2\}(\mathfrak{m}_2)$. Here, $\mathsf{SK} = \mathsf{SK}_{U_i}, \mathsf{ST}_1 =$ "$(\mathsf{PK}_{U_i}, \mathsf{SK}) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{PK}_{U_i}}$", $\mathsf{ST}_2 =$ "$(\mathsf{APK}, \mathsf{SK}) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$" and $\mathfrak{m}_2 = H_2(m, \rho, r_\rho, \chi, \mathsf{PK}_{U_i})$. Notice that U_i is able to calculate a correct η if χ is generated correctly. (In this case, ST_1 is true.)

   Due to the property of undeniable signatures, one is not able to verify χ with only public information. Thus, U_i needs to generate some kind of proof which can convince the verifier about the validity of χ. Instead of proving χ is a valid undeniable signature under PK_{U_i}, U_i proves that χ is valid either under PK_{U_i} or APK. This helps to conceal the identity of the full signature producer and make the arbitrator's involvement transparent.
6) $\mathsf{Ver}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK})$. Given a pair (m, σ) where $\sigma = (\rho, r_\rho, \chi, \eta)$ and two public keys $(\mathsf{PK}_{U_i}, \mathsf{APK})$, one can verify the correctness of ρ and η. If both are correct, this algorithm outputs "1." Otherwise, it outputs "0."

   This algorithm actually verifies two signatures: ρ and χ. The verification of ρ can be performed using U_i's public key, but the verification of the undeniable signature χ needs additional information, which is η in our protocol. The purpose of η is to prove that χ is a valid undeniable signature of PK_{U_i} or APK, and it can be independently generated by U_i (running the algorithm Sig) and arbitrator (running the algorithm Res).

7) $\mathsf{Res}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{ASK})$. For a resolution query $(m, \sigma^P, \mathsf{PK}_{U_i})$ from the verifier U_j, the arbitrator first checks if $\mathsf{PVer}(m, \sigma^P, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$ and U_j has fulfilled the obligation described in m. If so, the arbitrator will generate a full signature $(\rho, r_\rho, \chi, \eta)$, where $\rho = \sigma^P$, r_ρ is a random element in $\mathbb{Z}_{2^\kappa}$ and (χ, η) are calculated as follows:
   - The arbitrator uses the private key ASK to generate an undeniable signature χ on the message $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$. That is, $\chi = \mathsf{US.Sign}(\mathfrak{m}_1, \mathsf{ASK})$ and $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$.
   - The arbitrator then calculates the OR-proof $\eta = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_1 \vee \mathsf{ST}_2\}(\mathfrak{m}_2)$, where $\mathsf{SK} = \mathsf{ASK}$ and $(\mathsf{ST}_1, \mathsf{ST}_2, \mathfrak{m}_2)$ are the same as those in the algorithm Sig. Notice that the arbitrator is able to calculate a correct η if χ is generated correctly. (In this case, ST_2 is true.)

   As in the algorithm Sig, instead of proving χ is a valid undeniable signature under APK, the arbitrator proves that χ is valid either under PK_{U_i} or APK. This helps to conceal the identity of the full signature producer and make the arbitrator's involvement transparent.

8) $\mathsf{Prove}^{\mathsf{User}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{SK}_{U_i})$. Given a valid message–signature pair (m, σ) where $\sigma = (\rho, r_\rho, \chi, \eta)$, U_i calculates $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$ and runs the algorithm $\mathsf{US.Verify}(\mathfrak{m}_1, \chi, \mathsf{SK}_{U_i}, \mathsf{PK}_{U_i})$ to verify if $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{PK}_{U_i}}$. After that, U_i calculates $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$ and generates the proof as follows.
   - If $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{PK}_{U_i}}$, U_i can claim σ by generating $\pi^{\mathsf{User}}_{\mathsf{Claim}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_1\}(\mathfrak{m}_3)$. Here, $\mathsf{SK} = \mathsf{SK}_{U_i}$ and ST_1 is the same as that in Sig and Res.
   - Otherwise, U_i can deny σ by generating $\pi^{\mathsf{User}}_{\mathsf{Deny}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_3\}(\mathfrak{m}_3)$. Here, $\mathsf{SK} = \mathsf{SK}_{U_i}$ and $\mathsf{ST}_3 =$ "$(\mathsf{PK}_{U_i}, \mathsf{SK}) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \notin \mathcal{R}_{\mathsf{PK}_{U_i}}$".

9) $\mathsf{Prove}^{\mathsf{TTP}}(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{APK}, \mathsf{ASK})$. Given a valid message–signature pair (m, σ) where $\sigma = (\rho, r_\rho, \chi, \eta)$, the arbitrator calculates $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$ and runs the algorithm $\mathsf{US.Verify}(\mathfrak{m}_1, \chi, \mathsf{ASK}, \mathsf{APK})$ to verify if $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$. After that, the arbitrator calculates $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$ and generates the proof as follows.
   - If $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$, the arbitrator can claim σ by generating $\pi^{\mathsf{TTP}}_{\mathsf{Claim}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_2\}(\mathfrak{m}_3)$. Here, $\mathsf{SK} = \mathsf{ASK}$ and ST_2 is the same as that in Sig and Res.
   - Otherwise, the arbitrator can deny σ by generating $\pi^{\mathsf{TTP}}_{\mathsf{Deny}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_4\}(\mathfrak{m}_3)$. Here, $\mathsf{SK} = \mathsf{ASK}$ and $\mathsf{ST}_4 =$ "$(\mathsf{APK}, \mathsf{SK}) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \notin \mathcal{R}_{\mathsf{APK}}$".

10) Open. Given a valid message–signature pair (m, σ) under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and a proof π, one first calculates $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$.
   - If π is a correct SPK of the statement ST_1 or ST_4 on $\mathfrak{m}_3$, this algorithm outputs PK_{U_i};
   - Else, if π is a correct SPK of the statement ST_2 or ST_3 on $\mathfrak{m}_3$, this algorithm outputs APK; and
   - Otherwise, the output is "$\perp$."

The correctness of our construction is due to the correctness of the undeniable signature scheme US-Σ and the completeness of SPKs.

In Section IV-C, we provide an efficient instance of the proposed generic construction. The concrete protocol is based on the full-domain hash (FDH) [48], [49] variant of Chaum's undeniable signature scheme [42]. Section IV-D gives two variants of our generic construction, which address the issues of multiple arbitrators and designated verifier proofs.

B. Protocol Analysis

This subsection presents the analysis of the proposed generic protocol.

Efficiency Analysis:

1) Partial Signature: The partial signature in the proposed protocol is a normal digital signature such as Schnorr Signature [34]. Thus it requires almost the same computational and transmission costs as a normal digital signature.
2) Full Signature: The full signature of our protocol is made up of four parts: a partial signature, a random number in $\mathbb{Z}_{2^\kappa}$, an undeniable signature, and an OR-Signature. The computational and transmission costs associated with the first three parts are self-evident, but the last part, i.e., the OR-Signature, is a little bit more complex. Actually, the signing and verification of the OR-Signature are the most costly operations in the proposed protocol, and in a concrete instance (Section IV-C), it requires almost 5 times more computational cost and has a double signature size of the partial signature.

Security Analysis: The purpose of this paper is to formalize the notion of accountable OFE and show that there exists a construction of multiuser secure accountable OFE with a transparent third party. Thus the security analysis in this section consists of two parts. First, we will show that the proposed construction is multiuser secure under the security model defined in [2]. After that, we will prove (in the random oracle model) that our construction is an accountable OFE protocol with a transparent third party. However, the proposed construction may not have other properties required in specific applications. Nevertheless, we believe the results shown in this paper provide a feasible approach for the design of accountable OFE with those properties.

Security in the Multiuser Setting: As we introduced at the beginning of Section IV, our construction actually is a special case of the generic construction of multiuser secure OFE proposed in [2], namely the partial signature is an ordinary signature and the full signature consists of the partial signature and an OR-Signature. The OR-Signature can be generated by the signer or by the arbitrator. Thus, the security analysis of our construction in the multiuser setting is almost the same as that in [2], and the details are omitted here.

Accountability and Transparency: Before presenting the details, we first give a high-level illustration of how the accountability and the transparency are satisfied in our construction. The proposed construction is accountable since the undeniable signature χ contains the information of the actual full signature producer. However, it is computationally hard to tell who generates the full signature due to the anonymity of undeniable signatures. When needed, the signer or the arbitrator can publish a proof of the identity of the signature producer. This ensures that the arbitrator in our construction is transparent but responsible for its behaviors. The formal analysis consists of four theorems:

1) Theorem 1–Theorem 3 are devoted to showing that the proposed construction is accountable.
2) Theorem 4 is devoted to showing that the third party in our construction is transparent.

Theorem 1 (Type I Accountable): Our generic construction is $(t, q, \epsilon_{\rm US}^{\rm EU} + \epsilon_S + \epsilon_H)$-Type-I-Accountable assuming that the underlying undeniable signature scheme is $(t, q, \epsilon_{\rm US}^{\rm EU})$-Existentially Unforgeable, the SPK is (t, ϵ_S)-Sound, and the hash family $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant.

Proof: Let $\mathsf{OFE}\text{-}\mathcal{A}$ be the adversary who wishes to break the Type I accountability of our generic construction. At the end of the game, $\mathsf{OFE}\text{-}\mathcal{A}$ will output a message–signature pair $(\hat{m}, \hat{\sigma})$, where $\hat{\sigma} = (\hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$. Let $\hat{\mathfrak{m}}_1 = H_1(\hat{m}, \hat{\rho}, \hat{r}_\rho, \mathsf{PK}^*)$. Let E be the event that $\mathsf{OFE}\text{-}\mathcal{A}$ wins the game, i.e., $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{PK}^*, \mathsf{APK}, \hat{\pi}) = \mathsf{APK}$. We consider the following three cases:

C_1: no collision happens on H_1 and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{APK}}$;

C_2: no collision happens on H_1 and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{APK}}$;

C_3: there is at least one collision on H_1.

Thus, the probability that $\mathsf{OFE}\text{-}\mathcal{A}$ wins the game is

$$\begin{aligned}\Pr[E] &= \Pr[E \mid C_1]\Pr[C_1] + \Pr[E \mid C_2]\Pr[C_2] \\ &\quad + \Pr[E \mid C_3]\Pr[C_3] \\ &\le \Pr[C_1] + \Pr[E \mid C_2] + \Pr[C_3].\end{aligned}$$

We will show that (1) $\mathsf{OFE}\text{-}\mathcal{A}$ in **Case** C_1 can be converted to an algorithm $\mathsf{US}\text{-}\mathcal{A}$ who breaks the unforgeability of the undeniable signature scheme; (2) $\mathsf{OFE}\text{-}\mathcal{A}$ in **Case** $E \mid C_2$ can be converted to an algorithm SPK-$\mathcal{A}$ who breaks the soundness of SPK; and (3) $\mathsf{OFE}\text{-}\mathcal{A}$ in **Case** C_3 can be converted to an algorithm *Hash*-$\mathcal{A}$ who breaks the collision-resistance of $\mathcal{H}$.

Case C_1: At the beginning, $\mathsf{US}\text{-}\mathcal{A}$ obtains a public key PK from its challenger of the underlying undeniable signature scheme. $\mathsf{US}\text{-}\mathcal{A}$ sets the arbitrator's public key $\mathsf{APK} = \mathsf{PK}$ and sends APK to $\mathsf{OFE}\text{-}\mathcal{A}$. After that, $\mathsf{US}\text{-}\mathcal{A}$ runs the algorithm $\mathsf{Setup}^{\mathsf{User}}$ to answer $\mathsf{OFE}\text{-}\mathcal{A}$'s UserCreate queries. As $\mathsf{US}\text{-}\mathcal{A}$ has private keys of users created by itself, it can simulate oracles $\mathcal{O}^{\mathsf{Corruption}}, \mathcal{O}^{\mathsf{PSig}}$ and $\mathcal{O}^{\mathsf{Sig}}$ by running corresponding algorithms. Similarly, $\mathsf{US}\text{-}\mathcal{A}$ can reply Prove queries with the form $(\cdot, \cdot, \mathsf{PK}_{U_i}, \mathsf{User})$ if the user is created by itself. The simulation of random oracles can be done simply by answering randomly but consistently. Below, we show how $\mathsf{US}\text{-}\mathcal{A}$ can answer Res queries and Prove queries (with the form $(\cdot, \cdot, \cdot, \mathsf{TTP})$).

For a Res query $(m, \sigma^P, \mathsf{PK}_{U_i})$ where $\sigma^P = \rho$ is a valid partial signature of m, $\mathsf{US}\text{-}\mathcal{A}$ first chooses a random number $r_\rho \in \mathbb{Z}_{2^\kappa}$. Then, it computes $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$ and sends $\mathfrak{m}_1$ to its challenger as a US.Sign query. Let the response be χ, which is a valid undeniable signature of $\mathfrak{m}_1$ (that is, $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$). After that, $\mathsf{US}\text{-}\mathcal{A}$ computes $\mathfrak{m}_2$ as defined in our generic construction and generates the SPK η. While $\mathsf{US}\text{-}\mathcal{A}$ does not know the private key of APK, it is still able to generate η in the random oracle model using rewinding techniques. This is similar to the proof of Schnorr signature in the random oracle model [40], [41]. Finally, $\mathsf{US}\text{-}\mathcal{A}$ outputs the full signature $(\rho, r_\rho, \chi, \eta)$ as the answer.

For a Prove query with the form $(m, \sigma, \mathsf{PK}_{U_i}, \mathsf{TTP})$ where $\sigma = (\rho, r_\rho, \chi, \eta)$ is a valid full signature, $\mathsf{US}\text{-}\mathcal{A}$ first computes $\mathfrak{m}_1$ as defined in our construction. Then, it sends $(\mathfrak{m}_1, \chi)$ as a $\mathsf{Confirmation/Disvowal}$ query to its challenger. Once obtaining the response (which can show whether $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}}$ or not), $\mathsf{US}\text{-}\mathcal{A}$ generates the proof $\pi_{\mathsf{Claim}}^{\mathsf{TTP}}$ or $\pi_{\mathsf{Deny}}^{\mathsf{TTP}}$ in the random oracle model.

At the end of Phase 2, $\mathsf{OFE}\text{-}\mathcal{A}$ will generate the target user's public key PK^*. After that, $\mathsf{US}\text{-}\mathcal{A}$ will answer queries in the same way we described earlier. These queries must satisfy the restrictions in the game of Type I accountability.

This completes the description of how $\mathsf{US}\text{-}\mathcal{A}$ can correctly answer $\mathsf{OFE}\text{-}\mathcal{A}$'s queries in Phase 1, Phase 2, and Phase 3. Let $(\hat{m}, \hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$ be the output of $\mathsf{OFE}\text{-}\mathcal{A}$ in Phase 4. If the case C_1 happens, then $\hat{\mathfrak{m}}_1$ has not appeared as one of US.Sign queries (as no collision happens on H_1) and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{APK}}$. Then, $\mathsf{US}\text{-}\mathcal{A}$ can set $(\hat{\mathfrak{m}}_1, \hat{\chi})$ as a valid forgery of the undeniable signature scheme. If the underlying undeniable signature scheme is $(t, q, \epsilon_{\rm US}^{\rm EU})$-Existentially Unforgeable, then $\Pr[C_1] \le \epsilon_{\rm US}^{\rm EU}$.

Case $E \mid C_2$: We construct an algorithm SPK-$\mathcal{A}$ who can use $\mathsf{OFE}\text{-}\mathcal{A}$ to break the soundness of SPK. In the proof, SPK-$\mathcal{A}$ will generate all private–public key pairs and thus can answer all queries from $\mathsf{OFE}\text{-}\mathcal{A}$. If the case C_2 happens, then $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{APK}}$ but $\mathsf{OFE}\text{-}\mathcal{A}$ can output a full signature $\hat{\sigma} = (\hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$ and a proof $\hat{\pi}$ such that $\mathsf{Ver}(\hat{m}, \hat{\sigma}, \mathsf{APK}, \mathsf{PK}^*) = 1$ and $\mathsf{Open}(\hat{m}, \hat{\sigma}, \mathsf{APK}, \mathsf{PK}^*, \hat{\pi}) = \mathsf{APK}$. Then, either $\hat{\eta}$ or $\hat{\pi}$ breaks the soundness of SPK. If the SPK is (t, ϵ_S)-Sound, then $\Pr[E \mid C_2] \le \epsilon_S$.

Case C_3: If this case happens, we show that an $\mathsf{OFE}\text{-}\mathcal{A}$ can be converted to an algorithm *Hash*-$\mathcal{A}$ who can find a collision of H_1. In the proof, *Hash*-$\mathcal{A}$ generates all private–public key pairs and thus can answer all queries from $\mathsf{OFE}\text{-}\mathcal{A}$. At the end of the game, if the case C_3 happens, then there is a collision of H_1. *Hash*-$\mathcal{A}$ outputs this collision and breaks the collision-resistance of the hash function $H_1 \in \mathcal{H}$. If $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant, then $\Pr[C_3] \le \epsilon_H$.

This completes the analysis of each case. We have obtained the upper bounder of $\Pr[E]$, i.e., $\Pr[E] \le \Pr[C_1] + \Pr[E \mid C_2] + \Pr[C_3] \le \epsilon_{\rm US}^{\rm EU} + \epsilon_S + \epsilon_H$. Thus, our generic construction is $(t, q, \epsilon_{\rm US}^{\rm EU} + \epsilon_S + \epsilon_H)$-Type-I-Accountable, if the underlying undeniable signature scheme is $(t, q, \epsilon_{\rm US}^{\rm EU})$-Existentially Unforgeable, the SPK is (t, ϵ_S)-Sound and the hash family $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant. This completes the proof of Theorem 1. ■

Theorem 2 (Type II Accountable): Our generic construction is $(t, q, e \cdot q \cdot \epsilon_{\rm US}^{\rm EU} + \epsilon_S + \epsilon_H)$-Type-II-Accountable assuming that the underlying undeniable signature scheme is $(t, q, \epsilon_{\rm US}^{\rm EU})$-Existentially Unforgeable, the SPK is (t, ϵ_S)-Sound and the hash family $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant. Here, e is the base of the natural logarithm.

Proof: Let $\mathsf{OFE}\text{-}\mathcal{A}$ be the adversary who wishes to break the Type II accountability of our generic construction. At the end of the game, $\mathsf{OFE}\text{-}\mathcal{A}$ will output a message–signature pair $(\hat{m}, \hat{\sigma})$ and a public key PK^*, where $\hat{\sigma} = (\hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$. Let $\hat{\mathfrak{m}}_1 = H_1(\hat{m}, \hat{\rho}, \hat{r}_\rho, \mathsf{PK}^*)$. Let E be the event that $\mathsf{OFE}\text{-}\mathcal{A}$ wins the game. We now consider following three cases.

C_1: No collision happens on H_1 and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{PK}^*}$;

C_2: No collision happens on H_1 and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{PK}^*}$; and

C_3: There is at least one collision on H_1.

Thus, the probability that $\mathsf{OFE}\text{-}\mathcal{A}$ wins the game is

$$\begin{aligned}\Pr[E] &= \Pr[E \mid C_1]\Pr[C_1] + \Pr[E \mid C_2]\Pr[C_2] \\ &\quad + \Pr[E \mid C_3]\Pr[C_3] \\ &\le \Pr[C_1] + \Pr[E \mid C_2] + \Pr[C_3].\end{aligned}$$

The followings are similar to those of Theorem 1.

Case C_1: At the beginning, $\mathsf{US}\text{-}\mathcal{A}$ will obtain a public key PK from its challenger of the underlying undeniable signature scheme. $\mathsf{US}\text{-}\mathcal{A}$ will choose a random number j from 1 to q_{UC}, where q_{UC} is the number of queries $\mathsf{OFE}\text{-}\mathcal{A}$ issues to the oracle $\mathcal{O}^{\mathsf{UserCreate}}$. The simulation of $\mathcal{O}^{\mathsf{UserCreate}}$ is as follows. For the jth $\mathsf{UserCreate}$ query, $\mathsf{US}\text{-}\mathcal{A}$ sets $\mathsf{PK}_{U_j} = \mathsf{PK}$ and $\mathsf{SK}_{U_j} = \perp$. For other queries, $\mathsf{US}\text{-}\mathcal{A}$ responds by running the algorithm $\mathsf{Setup}^{\mathsf{User}}$. For each created user, $\mathsf{US}\text{-}\mathcal{A}$ updates the list $\mathcal{L}^{\mathsf{PK}}$ and returns the public key to $\mathsf{OFE}\text{-}\mathcal{A}$.

In some phase, $\mathsf{OFE}\text{-}\mathcal{A}$ will generate the third party's public key APK. The simulations of other oracles are quite similar to those in the proof of Theorem 1. In particular, $\mathsf{US}\text{-}\mathcal{A}$ generates U_j's partial signatures in the random oracle model and uses its own challenger to generate undeniable signatures of U_j (or, verify their validity). The only difference is that $\mathsf{US}\text{-}\mathcal{A}$ may abort during the simulation of $\mathcal{O}^{\mathsf{Corruption}}$, whose probability will be shown shortly.

If $\mathsf{US}\text{-}\mathcal{A}$ does not abort during the simulation, then $\mathsf{OFE}\text{-}\mathcal{A}$ will output a pair $(\hat{m}, \hat{\sigma})$ and a public key PK^*. $\mathsf{US}\text{-}\mathcal{A}$ will abort if $\mathsf{PK}^* \neq \mathsf{PK}_{U_j}$. We now compute the probability that $\mathsf{US}\text{-}\mathcal{A}$ does not abort: (1) The probability that $\mathsf{US}\text{-}\mathcal{A}$ does not abort during the simulation of $\mathcal{O}^{\mathsf{Corruption}}$ is $(1 - 1/q_{\mathsf{UC}})^{q_C} \ge 1/e$, where q_C is the number of queries made to $\mathcal{O}^{\mathsf{Corruption}}$ and e is the base of the natural logarithm. (2) In Phase 3, $\mathsf{PK}^* = \mathsf{PK}_{U_j}$. This happens with probability at least $1/q_{\mathsf{UC}} \ge 1/q$, where q is the number of all queries made by $\mathsf{OFE}\text{-}\mathcal{A}$. Therefore, $\mathsf{US}\text{-}\mathcal{A}$ will not abort during the game with probability at least $(1)/(e \cdot q)$.

Let $(\hat{m}, \hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$ be the output of $\mathsf{OFE}\text{-}\mathcal{A}$ in Phase 3. If the case C_1 happens, then $\hat{\mathfrak{m}}_1$ has not appeared as one of $\mathsf{US.Sign}$ queries and $(\hat{m}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{PK}^*}$. Therefore, if the underlying undeniable signature scheme is $(t, q, \epsilon_{\mathrm{US}}^{\mathrm{EU}})$-Existentially Unforgeable, then $\Pr[C_1] \le e \cdot q \cdot \epsilon_{\mathrm{US}}^{\mathrm{EU}}$.

Case $E \mid C_2$ and Case C_3: The analysis is the same as that in the proof of Theorem 1.

This completes the analysis of each case, and we have obtained the upper bounder of $\Pr[E]$, i.e., $\Pr[E] \le \Pr[C_1] + \Pr[E \mid C_2] + \Pr[C_3] \le e \cdot q \cdot \epsilon_{\mathrm{US}}^{\mathrm{EU}} + \epsilon_S + \epsilon_H$. Thus, our generic construction is $(t, q, e \cdot q \cdot \epsilon_{\mathrm{US}}^{\mathrm{EU}} + \epsilon_S + \epsilon_H)$-Type-II-Accountable, if the underlying undeniable signature scheme is $(t, q, \epsilon_{\mathrm{US}}^{\mathrm{EU}})$-Existentially Unforgeable the SPK is (t, ϵ_S)-Sound and $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant. This completes the proof of Theorem 2. ■

Theorem 3 (Type III Accountable): Our generic construction is $(t, \epsilon_{\mathrm{US}}^{\mathrm{CR}} + 3\epsilon_S)$-Type-III-Accountable assuming that the underlying undeniable signature scheme is $(t, \epsilon_{\mathrm{US}}^{\mathrm{CR}})$-Collision-Resistant and the SPK is (t, ϵ_S)-Sound.

Proof: Let $\mathsf{OFE}\text{-}\mathcal{A}$ be the adversary who generates two different public keys $(\mathsf{PK}^*, \mathsf{APK})$ and wishes to break the Type III accountability of our generic construction. $\mathsf{OFE}\text{-}\mathcal{A}$ will output $(\hat{m}, \hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$, together with two proofs $\hat{\pi}^{\mathsf{User}}$ and $\hat{\pi}^{\mathsf{TTP}}$. Let $\hat{\mathfrak{m}}_1 = H_1(\hat{m}, \hat{\rho}, \hat{r}_\rho, \mathsf{PK}^*)$. We consider the following four cases: (1) C_1: $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{PK}^*}$ and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{APK}}$; (2) C_2: $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{PK}^*}$ and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{APK}}$; (3) C_3: $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{PK}^*}$ and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{APK}}$; and (4) C_4: $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{PK}^*}$ and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{APK}}$. Let E be the event that $\mathsf{OFE}\text{-}\mathcal{A}$ wins the game, then

$$\Pr[E] \le \Pr[C_1] + \Pr[E \mid C_2] + \Pr[E \mid C_3] + \Pr[E \mid C_4].$$

CASE C_1: In this case, $\mathsf{OFE}\text{-}\mathcal{A}$ can be converted into an adversary $\mathsf{US}\text{-}\mathcal{A}$ who breaks the collision-resistance of the underlying undeniable signature scheme. To do that, $\mathsf{US}\text{-}\mathcal{A}$ only needs to set $(\mathsf{PK}^*, \mathsf{APK})$ and $(\hat{\mathfrak{m}}_1, \hat{\chi})$ as its own output. If the underlying undeniable signature scheme is $(t, \epsilon_{\mathrm{US}}^{\mathrm{CR}})$-Collision-Resistant, then $\Pr[C_1] \le \epsilon_{\mathrm{US}}^{\mathrm{CR}}$.

CASE $E \mid C_2$: In this case, $\mathsf{OFE}\text{-}\mathcal{A}$ can be converted to an algorithm who breaks the soundness of SPK. As $(\hat{\mathfrak{m}}_1, \hat{\chi}) \in \mathcal{R}_{\mathsf{PK}^*}$ and $(\hat{\mathfrak{m}}_1, \hat{\chi}) \notin \mathcal{R}_{\mathsf{APK}}$, either $\hat{\pi}^{\mathsf{User}}$ or $\hat{\pi}^{\mathsf{TTP}}$ breaks the soundness of the SPK. If the underlying SPK is (t, ϵ_S)-Sound, then $\Pr[C_2] \le \epsilon_S$.

CASE $E \mid C_3$ and $E \mid C_4$: Similarly, $\Pr[C_3] \le \epsilon_S$ and $\Pr[C_4] \le \epsilon_S$.

We have obtained the upper bounder of $\Pr[E]$, i.e., $\Pr[E] \le \epsilon_{\mathrm{US}}^{\mathrm{CR}} + 3\epsilon_S$. Thus, our generic construction is $(t, \epsilon_{\mathrm{US}}^{\mathrm{CR}} + 3\epsilon_S)$-Type-III-Accountable, if the underlying undeniable signature scheme is $(t, \epsilon_{\mathrm{US}}^{\mathrm{CR}})$-Collision-Resistant and the SPK is (t, ϵ_S)-Sound. ■

Theorem 4 (Transparent Third Party): The third party in our generic construction is $(t, q, \epsilon_H + q/2^\kappa + \epsilon_{\mathrm{SEU}} + e \cdot q \cdot \epsilon_{\mathrm{US}}^{\mathrm{A}})$-Transparent, assuming that the underlying hash family $\mathcal{H}$ is (t, ϵ_H)-Collision-Resistant, the SPK is $(t, q, \epsilon_{\mathrm{SEU}})$-Strongly-Existentially-Unforgeable and the undeniable signature scheme is $(t, q, \epsilon_{\mathrm{US}}^{\mathrm{A}})$-Anonymous. Here, e is the base of the natural logarithm and κ is system's security parameter.

Proof: Let $\mathsf{OFE}\text{-}\mathcal{A}$ be the adversary defined in Section II-D, who will provide $(\hat{m}, \hat{\sigma}^P, \mathsf{PK}^*)$ in Phase 2. Let $(\hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$ be the response from the challenger. We now define the following two events in the game.

C_1: At least one collision happens on H_1 or H_2; and

C_2: $\hat{r}_\rho$ is chosen by the challenger when answering queries to $\mathcal{O}^{\mathsf{Sig}}$ and $\mathcal{O}^{\mathsf{Res}}$.

Let $\overline{C_1}$ and $\overline{C_2}$ be the complementary events, respectively. Let E be the event that at the end of the game $\mathsf{OFE}\text{-}\mathcal{A}$ outputs a correct guess $b' = b$. Then, we have

$$\begin{aligned}\Pr[E] &= \Pr[E \wedge (C_1 \wedge C_2)] + \Pr[E \wedge (C_1 \wedge \overline{C_2})] \\ &\quad + \Pr[E \wedge (C_2 \wedge \overline{C_1})] + \Pr[E \wedge (\overline{C_1} \wedge \overline{C_2})] \\ &\le \Pr[C_1] + \Pr[C_2] + \Pr[E \mid (\overline{C_1} \wedge \overline{C_2})].\end{aligned}$$

We now show the upper bound of each probability.

First, we have $\Pr[C_1] \le \epsilon_H$ if the hash family is (t, ϵ_H)-Collision-Resistant. Second, $\Pr[C_2] \le q/2^\kappa$ as $\hat{r}_\rho$ is randomly chosen by the challenger between 1 and 2^κ, where κ is system's security parameter. It remains to consider the event $E \mid (\overline{C_1} \wedge \overline{C_2})$.

Letting $\mathcal{E}$ be the event that in the game, $\mathsf{OFE}\text{-}\mathcal{A}$ generates a valid message–signature pair of the SPK, which is not among those returned from the challenger. It follows that

$$\begin{aligned}&\Pr[E \mid (\overline{C_1} \wedge \overline{C_2})]\\&\quad= \Pr[(E \mid (\overline{C_1} \wedge \overline{C_2})) \wedge \mathcal{E}] + \Pr[(E \mid (\overline{C_1} \wedge \overline{C_2})) \wedge \overline{\mathcal{E}}]\\&\quad\leq \Pr[\mathcal{E}] + \Pr[(E \mid (\overline{C_1} \wedge \overline{C_2})) \mid \overline{\mathcal{E}}].\end{aligned}$$

If the underlying SPK is $(t, q, \epsilon_{\text{SEU}})$-Strongly-Existentially-Unforgeable, then $\Pr[\mathcal{E}] \leq \epsilon_{\text{SEU}}$. We now calculate the upper bounder of $\Pr[(E \mid (\overline{C_1} \wedge \overline{C_2})) \mid \overline{\mathcal{E}}]$. We show that if the event $(E \mid (\overline{C_1} \wedge \overline{C_2})) \mid \overline{\mathcal{E}}$ happens, $\mathsf{OFE}\text{-}\mathcal{A}$ can be converted to an algorithm $\mathsf{US}\text{-}\mathcal{A}$ who breaks the anonymity (as defined in Section III-C) of the underlying undeniable signature scheme. Let $(\mathsf{PK}_0, \mathsf{PK}_1)$ be two public keys given to $\mathsf{US}\text{-}\mathcal{A}$ in the game of anonymity in the undeniable signature scheme. In Phase 1, $\mathsf{US}\text{-}\mathcal{A}$ sets $\mathsf{APK} = \mathsf{PK}_1$ and sends it to $\mathsf{OFE}\text{-}\mathcal{A}$. Then, $\mathsf{US}\text{-}\mathcal{A}$ will choose a random number j from 1 to q_{UC}, where q_{UC} is the number of queries $\mathsf{OFE}\text{-}\mathcal{A}$ issues to the oracle $\mathcal{O}^{\mathsf{UserCreate}}$.

In Phase 2, the simulations of oracles are quite similar to those in the proof of Theorem 2. For the jth user create request, PK_{U_j} is set as PK_0. $\mathsf{US}\text{-}\mathcal{A}$ will generate SPKs related to PK_{U_j} or APK (whose private keys are unknown) in the random oracle model. For Sig or Res queries which require valid undeniable signatures of PK_{U_j} or APK, $\mathsf{US}\text{-}\mathcal{A}$ uses its challenger to calculate the responses and generates SPKs in the random oracle model. The response to Prove queries is the same as that in the proof of Theorem 1. $\mathsf{US}\text{-}\mathcal{A}$ could only fail in the simulation of $\mathcal{O}^{\mathsf{Corruption}}$ in this phase. As shown in the proof of Theorem 2, the probability that $\mathsf{US}\text{-}\mathcal{A}$ does not fail is at least $1/e$, where e is the base of the natural algorithm.

Let $(\hat{m}, \hat{\rho}, \mathsf{PK}^*)$ be the output of $\mathsf{OFE}\text{-}\mathcal{A}$ at the end of Phase 2, $\mathsf{US}\text{-}\mathcal{A}$ fails and aborts if $\mathsf{PK}^* \neq \mathsf{PK}_{U_j}$. Otherwise, $\mathsf{US}\text{-}\mathcal{A}$ chooses a random element $\hat{r}_\rho \in \mathbb{Z}_{2^\kappa}$ and computes $\hat{\mathfrak{m}}_1 = H_1(\hat{m}, \hat{\rho}, \hat{r}_\rho, \mathsf{PK}_{U_j})$. Under the conditions $\overline{C_1}$ and $\overline{C_2}$, $\hat{\mathfrak{m}}_1$ has not appeared before as one of $\mathsf{US.Sign}$ queries, and thus $\mathsf{US}\text{-}\mathcal{A}$ can set $\hat{\mathfrak{m}}_1$ as the challenge message of the undeniable signature scheme. After obtaining the response $\hat{\chi}$ (valid either under PK_{U_j} or APK), $\mathsf{US}\text{-}\mathcal{A}$ will generate the SPK $\hat{\eta}$ in the random oracle model. After that, the full signature $\hat{\sigma} = (\hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$ is given to $\mathsf{OFE}\text{-}\mathcal{A}$. $\mathsf{US}\text{-}\mathcal{A}$ will not abort if $\mathsf{PK}^* = \mathsf{PK}_{U_j}$, which happens with probability $1/q_{\mathsf{UC}} \geq 1/q$. ($q$ is the number of all queries made by $\mathsf{OFE}\text{-}\mathcal{A}$).

In Phase 4, $\mathsf{US}\text{-}\mathcal{A}$ simulates all oracles in the same way we described in Phase 2. As $\mathsf{US}\text{-}\mathcal{A}$ is not allowed to make $\mathsf{US.Sign}$ or $\mathsf{Confirmation/Disavowal}$ queries about $\hat{\mathfrak{m}}_1$, the simulation could fail if one of the following events happens: (1) In the simulation of $\mathcal{O}^{\mathsf{Sig}}$ or $\mathcal{O}^{\mathsf{Res}}$, $\mathsf{US}\text{-}\mathcal{A}$ chooses a random number r_ρ which is the same as the $\hat{r}_\rho$ in Phase 3. (2) In the simulation of $\mathcal{O}^{\mathsf{Sig}}$ or $\mathcal{O}^{\mathsf{Res}}$, $\mathsf{US}\text{-}\mathcal{A}$ chooses a random number $r_\rho \in \mathbb{Z}_{2^\kappa}$ different from $\hat{r}_\rho$ in Phase 3, but $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}) = \hat{\mathfrak{m}}_1$ for a query (m, ρ, PK). (3) In the simulation of $\mathcal{O}^{\mathsf{Prove}}$, $\mathsf{OFE}\text{-}\mathcal{A}$ makes a query $(m, \sigma, \mathsf{PK}, \cdot)$ where $\sigma = (\rho, r_\rho, \chi, \eta)$ is a valid full signature, $\sigma \neq \hat{\sigma}$ and $H_1(m, \rho, r_\rho, \mathsf{PK}) = \hat{\mathfrak{m}}_1$.

It is evident that under the conditions $\overline{C_1}$ and $\overline{C_2}$, the first two events will not happen. We only need to consider the last event, which requires $(m, \rho, r_\rho) = (\hat{m}, \hat{\rho}, \hat{r}_\rho)$ and $\mathsf{PK} = \mathsf{PK}_{U_j}$ under the condition $\overline{C_1}$. Thus, we have $(\chi, \eta) \neq (\hat{\chi}, \hat{\eta})$ as $(m, \rho, r_\rho, \chi, \eta) \neq (\hat{m}, \hat{\rho}, \hat{r}_\rho, \hat{\chi}, \hat{\eta})$. We divide the event $(\chi, \eta) \neq (\hat{\chi}, \hat{\eta})$ by two subevents: (A) $\chi \neq \hat{\chi}$ and (B) $\chi = \hat{\chi}$ and $\eta \neq \hat{\eta}$. Let us consider $\mathfrak{m}_2 = H_2(\hat{m}, \hat{\rho}, \hat{r}_\rho, \chi, \mathsf{PK}_{U_j})$. Under the conditions $\overline{C_1}$ and $\overline{C_2}$, $\mathfrak{m}_2$ will not be the one generated during the simulation of $\mathcal{O}^{\mathsf{Sig}}$ and $\mathcal{O}^{\mathsf{Res}}$, and $\mathfrak{m}_2 \neq \hat{\mathfrak{m}}_2$ if $\chi \neq \hat{\chi}$. Therefore, in this case, $\mathsf{OFE}\text{-}\mathcal{A}$ does not obtain any valid signatures of $\mathfrak{m}_2$, but can generate a valid signature η (SPK) of $\mathfrak{m}_2$. This breaks the unforgeability of the SPK. If $\chi = \hat{\chi}$ and $\eta \neq \hat{\eta}$, then $\mathfrak{m}_2 = \hat{\mathfrak{m}}_2$ and $\hat{\eta}$ is a new valid signature of the message $\hat{\mathfrak{m}}_2$, which breaks the strong unforgeability of the SPK. Under the condition $\overline{\mathcal{E}}$, neither A nor B would happen. Therefore, $\mathsf{US}\text{-}\mathcal{A}$ will not abort in Phase 4 under the conditions $\overline{C_1}, \overline{C_2}$ and $\overline{\mathcal{E}}$.

Let Abort be the event that $\mathsf{US}\text{-}\mathcal{A}$ aborts during the simulations in Phase 2 and Phase 3. If this event happens, $\mathsf{US}\text{-}\mathcal{A}$ will pick a random bit $b' \in \{0,1\}$ and outputs b' as the answer. Otherwise, the simulations in Phase 2 and Phase 3 will not fail (which happens with probability at least $1/(e \cdot q)$). Then, at the end of the game, $\mathsf{OFE}\text{-}\mathcal{A}$ will output a guess $b' \in \{0,1\}$. $\mathsf{US}\text{-}\mathcal{A}$ will set b' as its own guess. If the undeniable signature scheme is $(t, q, \epsilon_{\text{US}}^A)$-Anonymous, we have

$$\frac{1}{2}\Pr[\mathsf{Abort}] + \Pr\left[\left(E \mid (\overline{C_1} \wedge \overline{C_2})\right) \mid \overline{\mathcal{E}}\right] \Pr[\overline{\mathsf{Abort}}] \leq \epsilon_{\text{US}}^A + 1/2.$$

Thus, $\Pr[(E \mid (\overline{C_1} \wedge \overline{C_2})) \mid \overline{\mathcal{E}}] \leq e \cdot q \cdot \epsilon_{\text{US}}^A + 1/2$. It follows that $\Pr[E \mid (\overline{C_1} \wedge \overline{C_2})] \leq \epsilon_{\text{SEU}} + e \cdot q \cdot \epsilon_{\text{US}}^A + 1/2$. Together with the previous analysis, we have

$$\Pr[E] \leq \epsilon_H + q/2^\kappa + \epsilon_{\text{SEU}} + e \cdot q \cdot \epsilon_{\text{US}}^A + 1/2.$$

Therefore, the third party in our construction is $(t, q, \epsilon_H + q/2^\kappa + \epsilon_{\text{SEU}} + e \cdot q \cdot \epsilon_{\text{US}}^A)$-Transparent, if the hash function is (t, ϵ_H)-Collision-Resistant, the SPK is $(t, q, \epsilon_{\text{SEU}})$-Strongly-Existentially-Unforgeable and the undeniable signature scheme is $(t, q, \epsilon_{\text{US}}^A)$-Anonymous. This completes the proof of Theorem 4. ■

This completes the analysis of the proposed protocol. The next two subsections present a concrete instance and two variants of our generic protocol.

C. Concrete OFE Protocol With Accountability and Transparency

Let $\mathsf{US}\text{-}\Sigma$ be the FDH variant of Chaum's undeniable signature scheme with the parameter $\mathsf{US.Param} = (G, p, \kappa, g, H_{\mathsf{US}})$. The system parameter in the concrete protocol is $\mathsf{Param} = \{\mathsf{US.Param}, \mathsf{US}\text{–}\Sigma, H_1, H_2, H_3, h\}$, where (H_1, H_2, H_3) are the same as those in the generic construction in Section IV-A and h is another hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_p$. The details of each algorithm are given as follows.

1) $\mathsf{Setup}^{\mathsf{TTP}}(\mathsf{Param})$. The arbitrator chooses a random element $\mathsf{ASK} \in \mathbb{Z}_p$ and calculates $\mathsf{APK} = g^{\mathsf{ASK}}$.
2) $\mathsf{Setup}^{\mathsf{User}}(\mathsf{Param})$. The user U_i chooses a random element $\mathsf{SK}_{U_i} \in \mathbb{Z}_p$ and calculates $\mathsf{PK}_{U_i} = g^{\mathsf{SK}_{U_i}}$.
3) $\mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK})$. For a message m, the partial signature of U_i is $\rho = (\rho_1, \rho_2)$, where $\rho_1 = h(m, g^r)$, $\rho_2 = r - \rho_1 \mathsf{SK}_{U_i}$ and $r \in_R \mathbb{Z}_p$. (Actually, ρ is a Schnorr signature on m).

4) $\mathsf{PVer}(m, \rho, \mathsf{PK}_{U_i}, \mathsf{APK})$. This algorithm outputs "1" if $\rho_1 = h(m, g^{\rho_2}(\mathsf{PK}_{U_i})^{\rho_1})$. Otherwise, this algorithm outputs "0."
5) $\mathsf{Sig}(m, \mathsf{SK}_{U_i}, \rho, \mathsf{APK})$. This algorithm generates a full signature $\sigma = (\rho, r_\rho, \chi, \eta)$, where ρ is the partial signature of m and r_ρ is a random element in $\mathbb{Z}_{2^\kappa}$. χ and η are calculated as follows.
   - $\chi = H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{SK}_{U_i}}$, where $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$.
   - To generate η, U_i first calculates $\mathfrak{m}_2 = H_2(m, \rho, r_\rho, \chi, \mathsf{PK}_{U_i})$. Then, U_i chooses three random integers $\eta_3, \eta_4, v \in_R \mathbb{Z}_p$, and calculates $\eta_2 = h(\mathfrak{m}_2, g^v, H_{\mathsf{US}}(\mathfrak{m}_1)^v, g^{\eta_3}\mathsf{APK}^{\eta_4}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\eta_3}\chi^{\eta_4}) - \eta_4$ and $\eta_1 = v - \eta_2\mathsf{SK}_{U_i}$.
   - $\eta = (\eta_1, \eta_2, \eta_3, \eta_4)$ is an SPK on message $\mathfrak{m}_2$, which can prove χ is a valid undeniable signature of U_i or the arbitrator. One can make η strongly unforgeable using techniques in [27].

   The generation of η is based on the noninteractive zero-knowledge proof for Chaum's scheme reviewed in [45].
6) $\mathsf{Ver}(m, (\rho, r_\rho, \chi, \eta), \mathsf{PK}_{U_i}, \mathsf{APK})$. This algorithm outputs "1" if $\mathsf{PVer}(m, \rho, \mathsf{PK}_{U_i}, \mathsf{APK}) = 1$ and $\eta_2 + \eta_4 = h(\mathfrak{m}_2, g^{\eta_1}(\mathsf{PK}_{U_i})^{\eta_2}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\eta_1}\chi^{\eta_2}, g^{\eta_3}\mathsf{APK}^{\eta_4}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\eta_3}\chi^{\eta_4})$. Otherwise, this algorithm outputs "0." Here, $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$ and $\mathfrak{m}_2 = H_2(m, \rho, r_\rho, \chi, \mathsf{PK}_{U_i})$.
7) $\mathsf{Res}(m, \mathsf{PK}_{U_i}, \rho, \mathsf{ASK})$. This algorithm can also generate a full signature $\sigma = (\rho, r_\rho, \chi, \eta)$. Here, ρ is the partial signature of m and r_ρ is a random element in $\mathbb{Z}_{2^\kappa}$. χ and η are calculated as follows.
   - $\chi = H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{ASK}}$, where $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$.
   - The generation of η is quite similar to that in algorithm Sig. To generate η, the arbitrator first calculates $\mathfrak{m}_2 = H_2(m, \rho, r_\rho, \chi, \mathsf{PK}_{U_i})$. Then, the arbitrator chooses three integers $\eta_1, \eta_2, v \in_R \mathbb{Z}_p$, and calculates $\eta_4 = h(\mathfrak{m}_2, g^{\eta_1}(\mathsf{PK}_{U_i})^{\eta_2}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\eta_1}\chi^{\eta_2}, g^v, H_{\mathsf{US}}(\mathfrak{m}_1)^v) - \eta_2$ and $\eta_3 = v - \eta_4\mathsf{ASK}$. One can make $\eta = (\eta_1, \eta_2, \eta_3, \eta_4)$ strongly unforgeable using techniques in [27].
8) $\mathsf{Prove}^{\mathsf{User}}$. For a message m and its valid full signature $\sigma = (\rho, r_\rho, \chi, \eta)$ under $(\mathsf{PK}_{U_i}, \mathsf{APK})$, U_i computes $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$ and generates the proof as follows. The proof generation is also based on the noninteractive zero-knowledge proof for Chaum's scheme reviewed in [45].
   a) If $\chi = H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{SK}_{U_i}}$, U_i can prove this by generating $\pi^{\mathsf{User}}_{\mathsf{Claim}} = (\pi_1, \pi_2)$, where $\pi_1 = h(\mathfrak{m}_3, g^w, H_{\mathsf{US}}(\mathfrak{m}_1)^w)$, $\pi_2 = w - \pi_1\mathsf{SK}_{U_i}$ and w is a random element in $\mathbb{Z}_p$.
   b) Otherwise, U_i generates $\pi^{\mathsf{User}}_{\mathsf{Deny}} = (\pi_1, \pi_2, \pi_3, \pi_4)$, where $\pi_1 = (H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{SK}_{U_i}}/\chi)^w$, $\pi_2 = h(\mathfrak{m}_3, \pi_1, g^\alpha/\mathsf{PK}^\beta_{U_i}, H_{\mathsf{US}}(\mathfrak{m}_1)^\alpha/\chi^\beta)$, $\pi_3 = \beta + w\pi_2$ and $\pi_4 = \alpha + w\cdot\pi_2\cdot\mathsf{SK}_{U_i}$. Here, w, α, β are random elements in $\mathbb{Z}_p$.
9) $\mathsf{Prove}^{\mathsf{TTP}}$. For a message m and its valid full signature $\sigma = (\rho, r_\rho, \chi, \eta)$ under $(\mathsf{PK}_{U_i}, \mathsf{APK})$, the arbitrator calculates $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$.
   a) If $\chi = H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{ASK}}$, the arbitrator can prove this by generating $\pi^{\mathsf{TTP}}_{\mathsf{Claim}} = (\pi_1, \pi_2)$, where $\pi_1 = h(\mathfrak{m}_3, g^w, H_{\mathsf{US}}(\mathfrak{m}_1)^w), \pi_2 = w - \pi_1\mathsf{ASK}$ and w is a random element in $\mathbb{Z}_p$.
   b) Otherwise, the arbitrator generates $\pi^{\mathsf{TTP}}_{\mathsf{Deny}} = (\pi_1, \pi_2, \pi_3, \pi_4)$, where $\pi_1 = (H_{\mathsf{US}}(\mathfrak{m}_1)^{\mathsf{ASK}}/\chi)^w$, $\pi_2 = h(\mathfrak{m}_3, \pi_1, g^\alpha/\mathsf{APK}^\beta, H_{\mathsf{US}}(\mathfrak{m}_1)^\alpha/\chi^\beta)$, $\pi_3 = \beta + w\pi_2$ and $\pi_4 = \alpha + w\cdot\pi_2\cdot\mathsf{ASK}$. Here, w, α, β are three random elements in $\mathbb{Z}_p$.
10) Open: For a message m, its valid full signature $\sigma = (\rho, r_\rho, \chi, \eta)$ under $(\mathsf{PK}_{U_i}, \mathsf{APK})$ and a proof π, this algorithm works as follows.
    a) If $\pi = (\pi_1, \pi_2) \in \mathbb{Z}_p^2$:
       i) Output "PK_{U_i}" if $\pi_1 = h(\mathfrak{m}_3, g^{\pi_2}(\mathsf{PK}_{U_i})^{\pi_1}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\pi_2}\chi^{\pi_1})$.
       ii) Output "APK" if $\pi_1 = h(\mathfrak{m}_3, g^{\pi_2}\mathsf{APK}^{\pi_1}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\pi_2}\chi^{\pi_1})$.
    b) Else, if $\pi = (\pi_1, \pi_2, \pi_3, \pi_4) \in G \times \mathbb{Z}_p^3$:
       i) Output "PK_{U_i}" if $\pi_1 \neq 1$ and $\pi_2 = h(\mathfrak{m}_3, \pi_1, g^{\pi_4}/\mathsf{APK}^{\pi_3}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\pi_4}/(\chi^{\pi_3}\cdot\pi_1^{\pi_2}))$.
       ii) Output "APK" if $\pi_1 \neq 1$ and $\pi_2 = h(\mathfrak{m}_3, \pi_1, g^{\pi_4}/(\mathsf{PK}_{U_i})^{\pi_3}, H_{\mathsf{US}}(\mathfrak{m}_1)^{\pi_4}/(\chi^{\pi_3}\cdot\pi_1^{\pi_2}))$.
    c) Output "$\perp$" for all other cases.

Security of the Concrete Protocol

Accountability: Due to Theorem 1 and Theorem 2, the Type I and Type II accountability of the generic protocol require that the underlying undeniable signatures must be existentially unforgeable, the SPK must be sound and the hash functions must be collision resistant. As a concrete instance, the protocol described above is based on the FDH variant of Chaum's undeniable signature (whose unforeability is based on the computational Diffie–Hellman problem [45] in the random oracle model) and SPKs (which are variants of Σ-protocol and the soundness is based on the discrete logarithm in the finite group G). Thus, the Type I and Type II accountability of our concrete protocol can be reduced to the hardness of the computational Diffie–Hellman problem in the random oracle model. Similarly, due to Theorem 3, the Type III accountability of the concrete protocol can be reduced to the hardness of the discrete logarithm in the random oracle model.

Transparent Third Party: Due to Theorem 4, the transparency of the third party in our protocol requires that the underlying undeniable signatures must be anonymous, the SPK must be strongly unforgeable and the hash functions must be collision resistant. As a concrete instance, the protocol described above is based on the FDH variant of Chaum's undeniable signature (whose anonymity is based on the decisional Diffie–Hellman problem [45] in the random oracle model) and SPKs (which are variants of Σ-protocol and the strong unforgeability is based on the discrete logarithm in the finite group G using the method in [27]). Thus, our concrete protocol has a transparent third party in the random oracle model if the decisional Diffie–Hellman problem is hard in the finite group G.

D. Two Variants of Our Generic Construction

We give two variants of our generic construction.

Variant I: In OFE, the role of the third party (i.e., the arbitrator) is to answer resolution queries and convert partial signatures to full signatures. Like other systems with the third party, OFE is susceptible to the Denial-of-Service (known as DoS) attack on the arbitrator. A natural way to improve system's resilience to DoS attack is to employ two or more third parties on OFE, such that each third party is able to answer resolution queries independently. In this scenario, "accountability" seems indispensable since there are many entities having the ability to produce full signatures.

To ensure the accountability in OFE with multiple third parties (whose public keys are $\mathsf{APK}_1, \mathsf{APK}_2, \ldots,$ and APK_n), we only need to make slight changes to the generic construction in Section IV-A:

(1) Define the statement ST_2 as: $\mathsf{ST}_2^1 \vee \mathsf{ST}_2^2 \vee \cdots \vee \mathsf{ST}_2^n$, where

$$\mathsf{ST}_2^i : (\mathsf{SK}, \mathsf{APK}_i) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}_i}.$$

In this way, $\eta = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_1 \vee \mathsf{ST}_2\}(\mathfrak{m}_2)$ proves that χ is a valid undeniable signature under a public key from the set $\{\mathsf{PK}_{U_i}, \mathsf{APK}_1, \mathsf{APK}_2, \ldots, \mathsf{APK}_n\}$.

(2) Replace the algorithm $\mathsf{Prove}^{\mathsf{TTP}}$ with a new algorithm $\mathsf{Prove}^{\mathsf{TTP}_i}$, which is run by the ith arbitrator.

$\mathsf{Prove}^{\mathsf{TTP}_i}$: On input a valid message–signature pair (m, σ) where $\sigma = (\rho, r_\rho, \chi, \eta)$, the ith arbitrator first computes $\mathfrak{m}_1 = H_1(m, \rho, r_\rho, \mathsf{PK}_{U_i})$. Then, this arbitrator computes $\mathfrak{m}_3 = H_3(m, \sigma, \mathsf{PK}_{U_i})$ and generates the proof accordingly:

- If $(\mathfrak{m}_1, \chi) \in \mathcal{R}_{\mathsf{APK}_i}$, output $\pi_{\mathsf{Claim}}^{\mathsf{TTP}_i} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_2^i\}(\mathfrak{m}_3)$;
- Otherwise, output $\pi_{\mathsf{Deny}}^{\mathsf{TTP}_i} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_4^i\}(\mathfrak{m}_3)$, where

$$\mathsf{ST}_4^i : (\mathsf{SK}, \mathsf{APK}_i) \in \mathcal{R}_{\mathsf{Key}} \wedge (\mathfrak{m}_1, \chi) \notin \mathcal{R}_{\mathsf{APK}_i}.$$

(3) The output of the algorithm Open is a public key from the set $\{\mathsf{PK}_{U_i}, \mathsf{APK}_1, \mathsf{APK}_2, \cdots, \mathsf{APK}_n\}$ or the symbol "$\perp$".

Variant II: The second variant is motivated by the designated verifier proofs [50], where only the specified verifier can be convinced by the proof. By incorporating the designated verifier proofs in OFE, the signer and the arbitrator can prove to a designated verifier who is the full signature producer, but the verifier cannot use the proof to convince anyone else about the identity of full signature producer. To achieve this, we only need to modify the proposed construction (in Section IV-A) as follows:

1) Define a new statement $\mathsf{ST}_5 : (\mathsf{SK}, \mathsf{PK}_{U_j}) \in \mathcal{R}_{\mathsf{Key}}$, where PK_{U_j} is the public key of the designated verifier generated by the algorithm $\mathsf{Setup}^{\mathsf{User}}(\mathsf{Param})$.
2) In $\mathsf{Prove}^{\mathsf{User}}$, the signer U_i proves to the designated verifier U_j that
$$\pi_{\mathsf{Claim}}^{\mathsf{User}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_1 \vee \mathsf{ST}_5\}(\mathfrak{m}_3)$$
or
$$\pi_{\mathsf{Deny}}^{\mathsf{User}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_3 \vee \mathsf{ST}_5\}(\mathfrak{m}_3).$$
3) In $\mathsf{Prove}^{\mathsf{TTP}}$, the arbitrator proves to the designated verifier U_j that
$$\pi_{\mathsf{Claim}}^{\mathsf{TTP}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_2 \vee \mathsf{ST}_5\}(\mathfrak{m}_3)$$
or
$$\pi_{\mathsf{Deny}}^{\mathsf{TTP}} = \mathrm{SPK}\{\mathsf{SK} : \mathsf{ST}_4 \vee \mathsf{ST}_5\}(\mathfrak{m}_3).$$
4) The algorithm Open is only meaningful to the verifier U_j, who is able to generate all the above proofs.

V. Conclusion

We formalized the notion of accountable OFE, where both the signer and the third party are responsible for their behaviors. This not only is the first complete definition since its seminal introduction a decade ago but also provides a feasible approach for the design of accountable OFE with other properties. As an example, we proposed a generic (and also the first) design of OFE where the third party is transparent and accountable. The design is based on several well-studied cryptographic primitives and satisfies all security requirements defined in this paper. A concrete instance was also provided to demonstrate that the generic construction is very efficient to instantiate.

Our paper only makes the first step towards the formalization of accountable OFE with a transparent third party, and there are some issues that need further investigation. The three kinds of accountability defined in this paper only capture the basic requirements of accountable OFE, in the sense that each accountable OFE protocol must have those properties. There would be other specific requirements of accountability within concrete scenarios, and identifying those requirements is one of the future work directions. On the other hand, our protocol is only proved secure under the random oracle assumption. While random oracles have been widely used in security proofs, a provably secure protocol without random oracles is certainly more desirable.

References

[1] N. Asokan, M. Schunter, and M. Waidner, "Optimistic protocols for fair exchange," in *Proc. CCS'97*, 1997, pp. 7–17, ACM.

[2] Y. Dodis, P. J. Lee, and D. H. Yum, "Optimistic fair exchange in a multi-user setting," in *Proc. PKC'07*, 2007, vol. 4450, pp. 118–133, LNCS, Springer.

[3] Y. Dodis and L. Reyzin, "Breaking and repairing optimistic fair exchange from PODC 2003," in *Proc. 2003 ACM Workshop on Digital Rights Management*, 2003, pp. 47–54, ACM.

[4] 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 *Proc. CT-RSA'08*, 2008, vol. 4964, pp. 106–120, LNCS, Springer.

[5] O. Markowitch and S. Kremer, "An optimistic non-repudiation protocol with transparent trusted third party," in *Proc. ISC'01*, 2001, vol. 2200, pp. 363–378, LNCS, Springer.

[6] H. Zhu, W. Susilo, and Y. Mu, "Multi-party stand-alone and setup-free verifiably committed signatures," in *Proc. PKC'07*, 2007, vol. 4450, pp. 134–149, LNCS, Springer.

[7] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures (extended abstract)," in *Proc. Eurocrypt'98*, 1998, vol. 1403, pp. 591–606, LNCS, Springer.

[8] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 4, pp. 593–610, Apr. 2000.

[9] G. Ateniese, "Efficient verifiable encryption (and fair exchange) of digital signatures," in *Proc. CCS'99*, 1999, pp. 138–146, ACM.

[10] F. Bao, R. H. Deng, and W. Mao, "Efficient and practical fair exchange protocols with off-line TTP," in *Proc. 1998 IEEE Symp. Security and Privacy*, 1998, pp. 77–85, IEEE Computer Society.

[11] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. EUROCRYPT'03*, 2003, vol. 2656, pp. 416–432, LNCS, Springer.

[12] J. Camenisch and I. Damgård, "Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes," in *Proc. Asiacrypt'00*, 2000, vol. 1976, pp. 331–345, LNCS, Springer.

[13] J. M. Park, E. K. P. Chong, and H. J. Siegel, "Constructing fair-exchange protocols for e-commerce via distributed computation of RSA signatures," in *Proc. PODC'03*, 2003, pp. 172–181, ACM.

[14] J. A. Garay, M. Jakobsson, and P. MacKenzie, "Abuse-free optimistic contract signing," in *Proc. CRYPTO'99*, 1999, vol. LNCS, 1666, pp. 449–466, Springer.

[15] G. Wang, "An abuse-free fair contract signing protocol based on the RSA signature," in *Proc. 14th Int. Conf. World Wide Web*, 2005, pp. 412–421, ACM.

[16] N. Asokan, V. Shoup, and M. Waidner, "Asynchronous protocols for optimistic fair exchange," in *Proc. 1998 IEEE Symp. Security and Privacy*, 1998, pp. 86–99.

[17] X. Huang, Y. Mu, W. Susilo, W. Wu, and Y. Xiang, "Further observations on optimistic fair exchange protocols in the multi-user setting," in *Proc. PKC 2010*, 2010, vol. 6056, pp. 124–141, LNCS, Springer.

[18] J. Zhou, R. Deng, and F. Bao, "Some remarks on a fair exchange protocol," in *Proc. PKC'00*, 2000, vol. 1751, pp. 46–57, LNCS, Springer.

[19] J. Zhou and D. Gollmann, "A fair non-repudiation protocol," in *Proc. IEEE Symp. Security and Privacy 1996*, 1996, pp. 55–61, IEEE Computer Security Press.

[20] H. Zhu and F. Bao, "Stand-alone and setup-free verifiably committed signatures," in *Proc. CT-RSA'06*, vol. 3860, pp. 159–173, LNCS, Springer.

[21] H. Zhu and F. Bao, "More on stand-alone and setup-free verifiably committed signatures," in *Proc. ACISP'06*, 2006, vol. 4058, pp. 148–158, LNCS, Springer.

[22] Q. Huang, G. Yang, D. S. Wong, and W. Susilo, "Ambiguous optimistic fair exchange," in *Proc. Asiacrypt'08*, 2008, vol. 5350, pp. 74–89, LNCS, Springer.

[23] G. Ateniese and C. Nita-Rotaru, "Stateless-recipient certified e-mail system based on verifiable encryption," in *Proc. CT-RSA'02*, 2002, vol. 2271, pp. 182–199, LNCS, Springer.

[24] P. Liu, P. Ning, and S. Jajodia, "Avoiding loss of fairness owing to process crashes in fair data exchange protocols," *Decision Support Systems*, vol. 31, no. 3, pp. 337–350, 2001, Elsevier Science.

[25] R. Küesters, T. Truderung, and A. Vogt, Accountability: Definition and Relationship to Verifiability [Online]. Available: http://eprint.iacr.org/2010/236.pdf

[26] J. Onieva, J. Zhou, and J. Lopez, "Multi-party non-repudiation: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 1, Dec. 2008, ACM.

[27] Q. Huang, D. S. Wong, and Y. Zhao, "Generic transformation to strongly unforgeable signatures," in *Proc. ACNS'07*, 2007, vol. 4521, pp. 1–17, LNCS, Springer.

[28] D. Boneh, E. Shen, and B. Waters, "Strongly unforgeable signatures based on computational Diffie-Hellman," in *Proc. PKC'06*, 2006, vol. 3958, pp. 229–240, LNCS, Springer.

[29] R. Steinfeld, J. Pieprzyk, and H. Wang, "How to strengthen any weakly unforgeable signature into a strongly unforgeable signature," in *Proc. CT-RSA'07*, 2007, vol. 4377, pp. 357–371, LNCS, Springer.

[30] I. Teranishi, T. Oyama, and W. Oyama, "General conversion for obtaining strongly existentially unforgeable signatures," in *Proc. INDOCRYPT'06*, 2006, vol. 4329, pp. 191–205, LNCS, Springer.

[31] M. Bellare and O. Goldreich, "On defining proofs of knowledge," in *Proc. CRYPTO'92*, 1992, vol. 740, pp. 390–420, LNCS, Springer.

[32] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Proc. CRYPTO 1994*, 1994, vol. 839, pp. 174–187, LNCS, Springer.

[33] L. C. Guillou and J. J. Quisquater, "A "paradoxical" identity-based signature scheme resulting from zero-knowledge," in *Proc. CRYPTO'88*, 1990, vol. 403, pp. 216–231, LNCS, Springer.

[34] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Proc. CRYPTO'89*, 1990, vol. 435, pp. 239–252, LNCS, Springer.

[35] U. Feige and A. Shamir, "Zero knowledge proofs of knowledge in two rounds," in *Proc. CRYPTO'89*, 1990, vol. 435, pp. 526–544, LNCS, Springer.

[36] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems," *J. ACM*, vol. 38, no. 3, pp. 691–729, 1991.

[37] U. Feige and A. Shamir, "Witness indistinguishable and witness hiding protocols," in *Proc. 22nd STOC*, 1990, pp. 416–426, ACM.

[38] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proc. CRYPTO'86*, 1987, vol. 263, pp. 186–194, LNCS, Springer.

[39] J. Camenisch, "Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem," Ph.D. thesis, ETH Zürich, 1998, Diss. ETH No. 12520, Hartung Gorre Verlag, Konstanz.

[40] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proc. CCS'93*, 1993, pp. 62–73, ACM.

[41] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in *Proc. EUROCRYPT'96*, 1996, vol. 1070, pp. 387–398, LNCS, Springer.

[42] D. Chaum and H. V. Antwerpen, "Undeniable signatures," in *Proc. CRYPTO'89*, 1990, vol. 435, pp. 212–216, LNCS, Springer.

[43] X. Huang, Y. Mu, W. Susilo, and W. Wu, "Provably secure pairing-based convertible undeniable signature with short signature length," in *Proc. Pairing-Based Cryptography-Pairing*, 2007, vol. 4575, pp. 367–391, LNCS, Springer.

[44] F. Laguillaumie and D. Vergnaud, "Short undeniable signatures without random oracles: The missing link," in *Proc. Indocrypt'05*, 2005, vol. 3797, pp. 283–296, LNCS, Springer.

[45] W. Ogata, K. Kurosawa, and S.-H. Heng, "The security of the FDH variant of chaum undeniable signature scheme," in *Proc. PKC'05*, 2005, vol. 3386, pp. 328–345, LNCS, Springer.

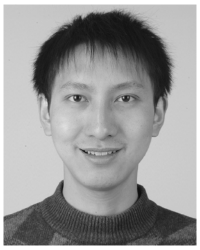
[46] J. Camenisch and V. Shoup, "Practical verifiable encryption and decryption of discrete logarithms," in *Proc. CRYPTO'03*, 2003, vol. 2729, pp. 126–144, LNCS, Springer.

[47] S. D. Galbraith and W. Mao, "Invisibility and anonymity of undeniable and confirmer signatures," in *Proc. CT-RSA'03*, 2003, vol. 2612, pp. 80–97, LNCS, Springer.

[48] M. Bellare and P. Rogaway, "The exact security of digital signatures-how to sign with RSA and Rabin," in *Proc. EUROCRYPT'96*, 1996, vol. 1070, pp. 399–416, LNCS, Springer.

[49] J. Coron, "On the exact security of full domain hash," in *Proc. CRYPTO'00*, 2000, vol. 1880, pp. 229–235, LNCS, Springer.

[50] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *Proc. Eurocrypt'96*, 1996, vol. 1070, pp. 143–154, LNCS, Springer.



Xinyi Huang received the Ph.D. degree in computer science (information security) from the School of Computer Science and Software Engineering, University of Wollongong, Australia, in 2009.

He is currently a research fellow at the Institute for Infocomm Research (I^2R), Singapore. His research interests focus on the cryptography and its applications in information systems. He has published more than 40 referred research papers at international conferences and journals.

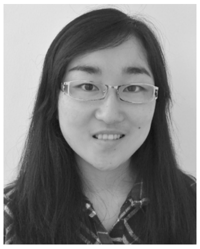
Yi Mu (M'03–SM'03) received the Ph.D. degree from the Australian National University in 1994.

He currently is an associate professor in the School of Computer Science and Software Engineering and the director of the Centre for Computer and Information Security Research, University of Wollongong, Wollongong, Australia. His current research interests include network security, computer security, and cryptography. He has published over 250 research papers.

Dr. Mu is the editor-in-chief of the *International Journal of Applied Cryptography* and serves as editor for nine other international journals. He is a member of the IACR.

Willy Susilo (M'06–SM'08) received the Ph.D. degree in computer science from University of Wollongong, Wollongong, Australia.

He is a Professor at the School of Computer Science and Software Engineering and the director of the Centre for Computer and Information Security Research (CCISR) at the University of Wollongong. He is currently holding the prestigious ARC Future Fellow awarded by the Australian Research Council (ARC). His main research interests include cryptography and information security. His main contribution is in the area of digital signature schemes, in particular fail-stop signature schemes and short signature schemes. He has served as a program committee member in dozens of international conferences. He has published numerous publications in the area of digital signature schemes and encryption schemes.

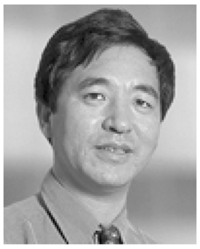


Wei Wu is currently working toward the Ph.D. degree at the School of Computer Science and Software Engineering, University of Wollongong, Wollongong, Australia.

Her research interests include new public key cryptography systems and secure server-aided computation. She has published more than 15 referred research papers at international conferences and journals.

Jianying Zhou received Ph.D. degree in information security from the University of London in 1997.

He is a senior scientist at the Institute for Infocomm Research (I^2R), Singapore, and heads the Network Security Group. His research interests are in computer and network security, cryptographic protocol, and mobile and wireless communications security. He has published about 150 referred papers at international conferences and journals. He is actively involved in the academic community, having served in many international conference committees as general chair, program chair, and PC member, having been in the editorial board and as a regular reviewer for many international journals. He is a cofounder and steering committee member of International Conference on Applied Cryptography and Network Security (ACNS).



Robert H. Deng (M'04–SM'09) received the Bachelor degree from the National University of Defense Technology, China, and the M.Sc. and Ph.D. degrees from the Illinois Institute of Technology.

He has been with the Singapore Management University since 2004, and is currently Professor, Associate Dean for Faculty and Research, School of Information Systems. Prior to this, he was Principal Scientist and Manager of Infocomm Security Department, Institute for Infocomm Research, Singapore. He has 26 patents and more than 200 technical publications in international conferences and journals in the areas of computer networks, network security, and information security. He has served as general chair, program committee chair, and program committee member of numerous international conferences. He is an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Associate Editor of *Security and Communication Networks Journal* (Wiley), and member of Editorial Board of the *Journal of Computer Science and Technology* (Chinese Academy of Sciences).

Dr. Deng received the University Outstanding Researcher Award from the National University of Singapore in 1999 and the Lee Kuan Yew Fellow for Research Excellence from the Singapore Management University in 2006. He was named Community Service Star and Showcased Senior Information Security Professional by $(ISC)^2$ under its Asia-Pacific Information Security Leadership Achievements program in 2010.