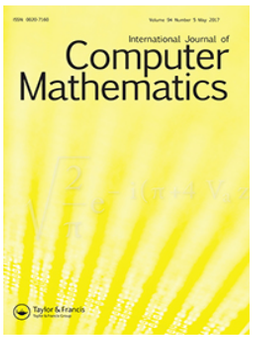

International Journal of Computer Mathematics

ISSN: 0020-7160 (Print) 1029-0265 (Online) Journal homepage: www.tandfonline.com/journals/gcom20

A new generic construction of anonymous designated confirmer signature for privacy-preserving fair exchange

Jiannan Wei, Guomin Yang & Yi Mu

To cite this article: Jiannan Wei, Guomin Yang & Yi Mu (2017) A new generic construction of anonymous designated confirmer signature for privacy-preserving fair exchange, International Journal of Computer Mathematics, 94:5, 946-961, DOI: 10.1080/00207160.2016.1154951

To link to this article: https://doi.org/10.1080/00207160.2016.1154951

Published online: 25 Mar 2016.

INTERNATIONAL JOURNAL OF COMPUTER MATHEMATICS, 2017
VOL. 94, NO. 5, 946–961
http://dx.doi.org/10.1080/00207160.2016.1154951

Taylor & Francis
Taylor & Francis Group

A new generic construction of anonymous designated confirmer signature for privacy-preserving fair exchange

Jiannan Wei, Guomin Yang and Yi Mu

Centre for Computer and Information Security Research, School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia

ABSTRACT

Designated confirmer signature (DCS), introduced by Chaum at *Eurocrypt* 1994, can be used to control the public verifiability of a digital signature. It is a very useful tool in many applications which require a signature to remain anonymous and unverifiable until a certain time/condition is reached. In this paper, we propose a new *generic* construction of Anonymous Identity-Based DCS from Anonymous Identity-Based Signature. Interestingly, our construction also automatically implies an Anonymous Identity-Based Convertible Undeniable Signature scheme. We prove that the construction is secure for signer, verifier and confirmer in the standard model.

ARTICLE HISTORY
Received 3 August 2015
Revised 9 November 2015
Accepted 23 December 2015

KEYWORDS
Identity-based cryptography; designated confirmer signature; anonymous signature; undeniable signature; fair exchange

2010 AMS SUBJECT CLASSIFICATIONS
94A60; 94A62

1. Introduction

Digital signature is a fundamental cryptographic primitive. A normal digital signature can be publicly verified by any entity once the message and the signer's public key are given. However, this feature may not be desirable in some applications such as fair exchange and fair contract signing. Consider the following scenario: Alice and Bob want to sign a contract. However, Alice does not want Bob to first obtain her signature since otherwise Bob can use it as a proof and bargain with another party, and vice versa. If the signatures are not publicly verifiable when they are exchanged, and later can be converted into publicly verifiable ones, then this problem can be solved.

One way to address this problem is to use an *undeniable signature*, introduced by Chaum [6] and Chaum and Antwerpen [8]. An undeniable signature ensures that the validity of a signature can only be verified with the help of the signer. A *convertible* undeniable signature scheme also allows the signer to convert the signature into a publicly verifiable one. However, in the above example, if a signer refuses to do the conversion, then the signature or contract will remain unverifiable.

In *Eurocrypt* 1994, Chaum [7] introduced the concept of *designated confirmer signature* (*DCS*). In DCS, a semi-trusted third party called the *designated confirmer* has the ability to verify the signature and to confirm valid signatures and disavow invalid signatures to any verifier. Also, similar to the convertible undeniable signature, a convertible DCS allows the confirmer to transform a valid DCS into a publicly verifiable standard signature. So in the above contract signing example, we can let the confirmer collect the DCS from each party, and then transform them into publicly verifiable ones after both signatures have been collected and verified.

CONTACT Jiannan Wei jw903@uowmail.edu.au Centre for Computer and Information Security Research, School of Computing and Information Technology, University of Wollongong, Wollongong, NSW 2522, Australia

Two basic requirements of a DCS scheme are *unforgeability* and *invisibility*. The former requires that only the real signer can produce a valid DCS; while the latter requires that no one except the confirmer (and the real signer) can determine whether a given message-signature pair is valid for a given user. In [10], Galbraith and Mao pointed out that '. . . anonymity rather than invisibility should be considered as the main security property for undeniable and confirmer signatures in the multiuser setting . . . '. They also provided a formal definition of *anonymity* for DCS schemes, which means no one except the confirmer (and the real signer) can tell the identity of the real signer. Interestingly, Galbraith and Mao showed that invisibility and anonymity are closely related by proving that these two notions are equivalent under some conditions.

Following Galbraith and Mao's observation cited above, we stress that the anonymity property of a DCS scheme is very important in many applications. Consider the contract signing example we gave above. If a DCS is not anonymous, then people can still learn the identity of the signer from the DCS even it is invisible. As a result, given a DCS from Bob, Alice can still prove to another party (e.g. a competitor of Bob) that Bob is negotiating a contract with her, and vice versa. Recently, Huang *et al.* [17] also showed that the anonymity property of a DCS is very important in their generic construction of an Ambiguous (or Privacy-Preserving) Optimistic Fair Exchange protocol [18].

Previous Work on DCS. Many DCS schemes [4,11,13,16,20–22,25,26,28] have been proposed since the seminal work by Chaum [7]. Below we briefly review these constructions.

Okamoto [22] proposed the first formal model for DCS, and proved that DCS is equivalent to public key encryption. However, Michels and Stadler [20] showed that Okamoto's scheme has the problem that the confirmer can forge a valid signature of a signer. They also proposed a new construction which uses the Commit-then-Sign approach where the signer signs a commitment of the message, and the commitment can be opened by the confirmer.

In [4], Camenisch and Michels proposed the idea of Sign-then-Encrypt for constructing DCS schemes. The idea is to encrypt a signature under the confirmer's public key. The main difficulty of implementing this approach is to avoid the generic zero-knowledge proofs in the confirm and disavowal protocols. In [13], Goldwasser and Waisbard proposed to use *strong witness hiding proofs of knowledge* to implement the confirm protocol. However, general zero-knowledge proof is still required in the disavowal protocol. Later, Gentry, Molnar and Ramzan (GMR) [11] proposed a DCS scheme based on verifiable public-key encryption [5]. Their scheme does not require generic zero-knowledge proof in the confirm or disavowal protocol. However, in [25], Wang *et al.* demonstrated two security issues related to the unfoolability and invisibility of the GMR scheme.

We should note that many generic DCS constructions (e.g. [11,20,25]) can provide invisibility but not anonymity. The reason is that in these constructions an ordinary signature of the signer and the content (e.g. a commitment of the message) being signed are both included in a DCS. It is obvious that by verifying the ordinary signature, the verifier can easily tell the identity of the signer. Therefore, these DCS schemes cannot be used to build privacy-preserving (or ambiguous) fair exchange protocols.

1.1. Motivation and our contributions

In contrast, the notion of 'anonymous signature' was proposed by Yang *et al.* [27] and further studied in [9,29]. An anonymous signature is an ordinary signature with the property of signer anonymity, that is no one is able to recognize the signer's identity given only the signature but not the message being signed. This feature is very similar to the invisibility and anonymity of a DCS. Such an observation, together with the fact that many of the previous generic DCS constructions cannot achieve anonymity, motivated us to explore new solutions for constructing anonymous DCS schemes.

The main contribution of this paper is to provide a new approach for the construction of *anonymous* DCS schemes. We demonstrate a new systematic way to build anonymous DCS from anonymous signature. The previous Sign-then-Encrypt approach [4] utilizes public key encryption to achieve invisibility and anonymity, in this paper, we will tackle the problem from a different angle.

Instead of encrypting the signature, we make the message invisible by embedding some secret information into it, and then apply an anonymous signing algorithm to produce the final signature. By ensuring that only the designated confirmer can recover the secret embedded in the message, we can achieve invisibility and anonymity based on the security of the anonymous signature. We prove the security of our new construction and also provide some guidelines on the instantiation of our generic framework.

As demonstrated by Camenisch and Michels [4], a DCS secure in the single-user/single-confirmer setting may not be secure in the multi-user/single-confirmer setting. In this paper, we consider DCS in the identity-based setting [24], which is a more general multi-user/multi-confirmer setting where each signer can also be a confirmer. Interestingly, in the special case that a signer chooses him/herself as the confirmer, our identity-based DCS scheme automatically becomes a convertible identity-based undeniable signature scheme.

1.2. Paper organization

In Section 2, some basic primitives used in our construction are reviewed. In Section 3, we present the definitions and security models for ID-based signature (IBS) and ID-based designated confirmer signature (IBDCS). We then show our generic anonymous IBDCS scheme in Section 4 and prove its security in Section 5. We present some guidelines on the instantiation of the proposed generic scheme in Section 6. Some applications of the proposed IBDCS scheme are presented in Section 7, and the paper is concluded in Section 8.

2. Preliminaries

In this section, we first introduce some cryptographic primitives that will be used in our construction.

2.1. ID-based one-way key exchange

An ID-based one-way key exchange (IDOWKE) protocol consists of the following polynomial-time algorithms:

- $Setup(1^k)$: it takes the security parameter 1^k as input and outputs the system parameters *param*, a master public key *mpk*, and the corresponding master secret key *msk*.
- $Extract(msk, \mathrm{ID})$: the key extraction algorithm takes as input *msk* and a user identity, and outputs a user secret key SK_{ID}.
- $KI(\mathrm{ID}_B)$: the initiator A of the protocol takes as input a responder's identity ID_B, and outputs a message M_{AB} and a shared key K_{AB}.
- $KR(SK_{\mathrm{ID}_B}, M_{AB})$: the responder B of the protocol takes as input SK_{ID_B}, M_{AB}, and outputs a shared key K_{BA}.

An IDOWKE protocol should satisfy the following properties:

- Completeness: If A and B execute the protocol honestly, then $K_{AB} = K_{BA}$;
- Key privacy: consider the following game between a challenger $\mathcal{S}$ and an adversary $\mathcal{A}$.
  (1) Setup: $\mathcal{S}$ runs $Setup(1^k)$, and forwards *params* and *mpk* to $\mathcal{A}$.
  (2) Learning: $\mathcal{A}$ can adaptively issue the following queries
    - Extract: $\mathcal{A}$ provides an identity ID_i. $\mathcal{S}$ runs $Extract(msk, \mathrm{ID}_i)$ and returns the resulting private key SK_{ID_i} to $\mathcal{A}$.
    - Execute: $\mathcal{A}$ provides an identity ID_B. $\mathcal{S}$ runs $(M_{AB}, K_{AB}) \leftarrow KI(\mathrm{ID}_B)$ and returns M_{AB} to $\mathcal{A}$.

- Send: $\mathcal{A}$ provides an identity ID_B and a message M_{AB}. $\mathcal{S}$ runs $K_{AB} \leftarrow KR(SK_{\mathrm{ID}_B}, M_{AB})$ and returns '1' to the adversary.
- Reveal: $\mathcal{A}$ can learn the shared key K_{AB} for any session (ID_B, M_{AB}) created in an Execute or Send query.

(3) Challenge: $\mathcal{A}$ outputs ID^* which has never appeared in the Extract queries. $\mathcal{S}$ runs $(M^*, K^*) \leftarrow KI(\mathrm{ID}^*)$ and tosses a random coin b. If $b = 1$, then (M^*, K^*) is returned to $\mathcal{A}$; otherwise, (M^*, R) is returned to $\mathcal{A}$ where R denotes a randomly selected key. $\mathcal{A}$ can continue to make queries except that ID^* cannot appear in any Extract query and (ID^*, M^*) cannot appear in a Send query.

(4) Output: $\mathcal{A}$ outputs a bit b' as her guess for b.

Define the advantage of the adversary as

$$\mathrm{Adv}_{\mathcal{A}}^{KP}(k) = \Pr[b' = b] - \frac{1}{2}.$$

We say an IDOWKE protocol has key privacy if $\mathrm{Adv}_{\mathcal{A}}^{KP}(k)$ is negligible in k for any PPT adversary $\mathcal{A}$.

It is worth noting that we can build an IDOWKE protocol with key privacy from an ID-based Key Encapsulation Mechanism (IDKEM) that is IND-ID-CCA secure.

2.2. Zero knowledge proof system

A pair of interactive machines (P, V) is called an *interactive zero knowledge proof system* for an NP language L if machine V is polynomial-time bounded and the following three conditions hold

- Completeness: For every $x \in L$

$$\Pr(\langle P, V\rangle(x) = 1) = 1.$$

- Soundness: For every $x \notin L$ and every interactive machine B

$$\Pr(\langle B, V\rangle(x) = 1) \leq \mathrm{neg}(k),$$

where k denotes the system parameter and $neg(\cdot)$ is a negligible function.

- Zero-knowledge: Let $\mathrm{view}_{V^*}^{P(\omega)}(x)$ denote a random variable describing the contents of the random tape of V^* and the messages V^* receives from P during an execution of the protocol on common input $x \in L$, where P has auxiliary input ω which is the witness of x. We say that $\langle P, V\rangle$ is zero-knowledge if for every probabilistic polynomial-time (PPT) interactive machine V^* there exists a PPT simulator S which has oracle access to V^*, such that for all sufficiently long $x \in L$ the ensembles $\{\mathrm{view}_{V^*}^{P(\omega)}(x)\}_{x\in L}$ and $\{S^{V^*}(x)\}_{x\in L}$ are computationally indistinguishable. That is,

$$\mathrm{Adv}_{\mathcal{D}}^{zk}(k) = |\Pr[\mathcal{D}(\mathrm{view}_{V^*}^{P(\omega)}(x)) = 1] - \Pr[\mathcal{D}(S^{V^*}(x)) = 1]|$$

is a negligible function of the security parameter k for any PPT distinguisher $\mathcal{D}$.

2.3. Trapdoor hash function

A trapdoor hash function consists of a pair of algorithms:

- $KG(1^k)$: it is a PPT algorithm that on input 1^k outputs a key pair (HK, TK).
- $H_{HK}(M; R)$: each HK defines a randomized hash function that on input a message M and random coins R outputs a hash value.

A trapdoor hash function has the following properties:

(1) Collision resistance : There is no PPT algorithm $\mathcal{A}$ that on input HK can output, with a non-negligible probability, two pairs $(M_1, R_1), (M_2, R_2)$ that satisfy $M_1 \neq M_2$ and $H_{HK}(M_1; R_1) = H_{HK}(M_2; R_2)$. We define the advantage of the adversary as $\mathrm{Adv}^{cr}_{\mathcal{A}}(k) = \Pr[\mathcal{A}$ finds a collision].
(2) Trapdoor collision : There exists a polynomial-time algorithm that given (HK, TK), (M_1, R_1), and an additional message M_2, outputs a value R_2 such that $H_{HK}(M_1; R_1) = H_{HK}(M_2; R_2)$.
(3) Inversion: for any valid hash key HK, any message M, and any hash value h in the range of H_{HK}, there exists an R such that $H(M; R) = h$. Moreover, if h is selected uniformly at random, then R is also uniformly distributed.

3. Security definitions of anonymous IBS and IBDCS

3.1. IBS – definition

An IBS scheme consists of the following polynomial-time algorithms:

- $\mathcal{ST}(1^k)$: it takes the security parameter 1^k as input, and outputs the public parameters *params*, master public key *mpk* and master private key *msk*.
- $\mathcal{ET}(msk, \mathrm{ID})$: it takes master secret key *msk* and a user's identity ID as inputs, and outputs a user's secret key SK_{ID}.
- $\mathcal{SG}(SK_{\mathrm{ID}}, M)$: it takes a user secret key SK_{ID} and a message M as input, and outputs a signature σ.
- $\mathcal{VF}(mpk, \mathrm{ID}, M, \sigma)$: it takes the master public key *mpk*, a user identity ID, a message M, and a signature σ as input, and outputs either 0 (denoting 'reject') or 1 (denoting 'accept').

3.2. IBS security models

3.2.1. Unforgeability

We say an IBS scheme is *existentially unforgeable under adaptive chosen message and ID attacks* (UF-CMIDA) if no polynomial time algorithm $\mathcal{A}$ has a non-negligible advantage in the following game simulated by a challenger $\mathcal{C}$.

(1) $\mathcal{C}$ runs $\mathcal{ST}(1^k)$ to generate *param*, (mpk, msk), and passes *param* and *mpk* to $\mathcal{A}$.
(2) $\mathcal{A}$ can adaptively issue the following queries:
    - *Extract* query. on receiving an identity ID from $\mathcal{A}$, $\mathcal{C}$ sends back the private key associated to ID which is obtained by running $\mathcal{ET}(msk, \mathrm{ID})$.
    - *Signing* query. Given an identity ID and a message m, $\mathcal{C}$ first generates SK_{ID} (if the secret key for ID has never been generated before), and returns a signature which is generated by running $\mathcal{SG}(SK_{\mathrm{ID}}, m)$.
(3) $\mathcal{A}$ outputs $(\mathrm{ID}^*, m^*, \sigma^*)$, such that ID^* has not appeared in an *Extract* query, and (ID^*, m^*) has not appeared in a *Signing* query. We say $\mathcal{A}$ wins the game if σ^* is a valid signature w.r.t. m^* and ID^*.

$\mathcal{A}$'s advantage is defined as

$$\mathrm{Adv}^{\mathrm{uf-cmida}}_{\mathcal{A}}(k) = \Pr[\mathcal{A} \text{ wins}].$$

3.2.2. Signer anonymity

We say an IBS scheme has *signer anonymity under adaptive chosen message and ID attacks* (SA-CMIDA) if no polynomial time algorithm $\mathcal{D}$ can win the following game simulated by a simulator $\mathcal{C}$ with a probability significantly lager than 1/2.

(1) $\mathcal{C}$ runs $\mathcal{ST}(1^k)$ to generate *param*, (mpk, msk), and passes *param* and *mpk* to $\mathcal{D}$.
(2) $\mathcal{D}$ can adaptively make *Extract* and *Signing* queries as in the unforgeability game.
(3) $\mathcal{D}$ picks two identities $\mathrm{ID}_0^*, \mathrm{ID}_1^*$ which have not appeared in any *Extract* query.
(4) $\mathcal{C}$ tosses a random coin $\varpi \xleftarrow{R} \{0,1\}$, then uniformly picks a message m^* in the message space, and returns a challenge signature $\sigma^* \leftarrow \mathcal{SG}(SK_{\mathrm{ID}^*_\varpi})$ to $\mathcal{D}$. $\mathcal{D}$ can still adaptively make Extract and Signing queries, except that ID_0^* and ID_1^* cannot appear in any Extract query.
(5) $\mathcal{D}$ output a bit ϖ' and the adversary wins the game if $\varpi' = \varpi$.

The advantage of the adversary $\mathcal{D}$ is defined as:

$$\mathrm{Adv}_{\mathcal{D}}^{\mathrm{sa-cmida}}(k) = \Pr[\varpi' = \varpi] - \tfrac{1}{2}.$$

3.3. IBDCS – definition

An IBDCS scheme consists of the following polynomial-time algorithms:

- The setup ($\mathcal{ST}'$), extract ($\mathcal{ET}'$), sign ($\mathcal{SG}'$), and verify ($\mathcal{VF}'$) algorithms are the same as those in the IBS definition.
- $\mathcal{CS}'_{(S,V)}(m, \mathrm{ID}_S, \mathrm{ID}_C)$: the ConfirmedSign algorithm is an interactive protocol between a signer S (with private input SK_{ID_S}) and a verifier V. The common input is $(m, \mathrm{ID}_S, \mathrm{ID}_C)$ where ID_C denotes the identity of the designated confirmer. The output of V is a pair (b, σ') where $b \in \{0,1\}$ and σ' is a DCS on message m.
- $\mathcal{CF}'_{(C,V)}(m, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$: the Confirm algorithm is an interactive protocol between the confirmer C (with private input SK_{ID_C}) and a verifier V. The common input is $(m, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$, and the output of the protocol is $b \in \{0,1\}$.
- $\mathcal{DV}'_{(C,V)}(m, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$: the Disavowal algorithm is another interactive protocol between the confirmer C and a verifier V. The common input is $(m, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$, and C's private input is SK_{ID_C}, while the output of the protocol is $b \in \{0,1\}$.
- $\mathcal{ES}'(m, \sigma', SK_{\mathrm{ID}_C}, \mathrm{ID}_S)$: the ExtractSignature algorithm takes a message m, a DCS σ', the confirmer's secret key SK_{ID_C}, and the signer's identity ID_S as input, and outputs either a signature σ such that

$$\mathcal{VF}'(mpk, \mathrm{ID}_S, m, \sigma) = 1$$

or a special symbol $\perp$.

3.4. IBDCS security models

3.4.1. Unforgeability: security for signer

We say an IBDCS scheme has *existential unforgeability under adaptive chosen message and* ID *attacks* (UF-CMIDA) if no PPT adversary can forge a valid DCS on a message m, even if the adversary knows the confirmer's secret key. This is captured in the following game between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$.

- Setup Phase: $\mathcal{S}$ runs the setup algorithm to generate *param*, (mpk, msk), and passes *param* and *mpk* to $\mathcal{A}$.
- Training Phase:
  - Extract Query: same as the Extract Query defined in the IBS unforgeability game.
  - ConfirmedSign Query: $\mathcal{A}$ provides a signer identity ID_S, a confirmer identity ID_C, and a message m to $\mathcal{S}$, who then runs the ConfirmedSign protocol $(\mathcal{CS}'_{(S,\mathcal{A})})$ with $\mathcal{A}$ using SK_{ID_S}.
  - Confirm Query: $\mathcal{A}$ can request to verify a message-DCS pair (m, σ') w.r.t. a signer ID_S and a designated confirmer ID_C. $\mathcal{S}$ runs the Confirm protocol $(\mathcal{CF}'_{(C,\mathcal{A})})$ with $\mathcal{A}$ using SK_{ID_C}.

- Disavowal Query: similar to the Confirm query, $\mathcal{S}$ runs the Disavowal protocol $(\mathcal{DV}'_{(C,\mathcal{A})})$ with $\mathcal{A}$ using SK_{ID_C}.
- ExtractSignature Query: $\mathcal{A}$ can request to extract a signature from a DCS by providing a message m, a DCS σ', a confirmer's identity ID_C, and a signer's identity ID_S. The challenger $\mathcal{S}$ runs $\mathcal{ES}'(m, \sigma', SK_{\mathrm{ID}_C}, \mathrm{ID}_S)$ and returns the result back to $\mathcal{A}$.

- Output: $\mathcal{A}$ outputs a message-DCS pair (m^*, σ'^*) for signer ID_S^* and designated confirmer ID_C^* such that
  - ID_S^* has not appeared in an Extract query;
  - (m^*, ID_S^*) has not appeared in a ConfirmedSign query.

We say $\mathcal{A}$ wins the unforgeability game if

$$\mathcal{VF}'(mpk, \mathrm{ID}_S^*, m^*, \mathcal{ES}'(m^*, \sigma'^*, SK_{\mathrm{ID}_C^*}, \mathrm{ID}_S^*)) = 1.$$

The advantage of $\mathcal{A}$ is defined as:

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{UF-CMIDA}}(k) = \Pr[\mathcal{A} \text{ wins}].$$

3.4.2. Anonymity: security for signer

Anonymity for signer means given a DCS, a verifier (without the help of the designated confirmer) should not be able to tell the identity of the signer.

- Setup and Training Phases: same as in the unforgeability game.
- Key Selection Phase: $\mathcal{A}$ picks a message m^*, two signers' identities $\mathrm{ID}_{S_0}^*, \mathrm{ID}_{S_1}^*$ and a confirmer's identity ID_C^*. It is required that$\mathrm{ID}_{S_0}^*$ or $\mathrm{ID}_{S_1}^*$ or ID_C^* has never appeared in an Extract query.
- Challenge Phase: $\mathcal{S}$ tosses a random coin b and computes

$$(1, \sigma'^*) \leftarrow \mathcal{CS}'_{(\mathrm{ID}_{S_b}^*, V)}(m^*, \mathrm{ID}_{S_b}^*, \mathrm{ID}_C^*),$$

and returns σ'^* to $\mathcal{A}$. $\mathcal{A}$ can continue to make queries as in the Training Phase, except that
  - $\mathrm{ID}_{S_0}^*$ or $\mathrm{ID}_{S_1}^*$ or ID_C^* has never appeared in an Extract query; and
  - σ'^* has appeared in any Confirm/Disavowal/ExtractSignature query.
- Output Phase: $\mathcal{A}$ outputs a bit b' and the adversary wins the game if $b' = b$.

The adversary $\mathcal{A}$'s advantage is defined as:

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{SA-CMIDA}}(k) = |\Pr[b' = b] - \tfrac{1}{2}|.$$

3.4.3. Invisibility: security for confirmer

Invisibility means without the help from either the signer or the designated confirmer, an adversary cannot determine the validity of a DCS.

- Setup and Training Phases: same as in the unforgeability game.
- Challenge Phase: The adversary $\mathcal{A}$ outputs a messages m^*, a signer's identity ID_S^* and a designated confirmer's identity ID_C^* with the restriction that ID_S^* or ID_C^* has not appeared in Extract queries. The challenger $\mathcal{C}$ tosses a random coin $b \in \{0, 1\}$. If $b = 0$, $\mathcal{C}$ computes a valid DCS signature σ'^* using m^*, $SK_{\mathrm{ID}_S^*}$ and ID_C^*. Otherwise, σ'^* is chosen uniformly at random from the DCS space of ID_S^*. After receiving σ'^*, $\mathcal{A}$ can continue to make queries as in the Training Phase except that σ'^* cannot appear in any query.
- Output Phase: $\mathcal{A}$ outputs a bit b'.

We say the adversary $\mathcal{A}$ wins the game if $b' = b$. The advantage of the adversary is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{invis}}(k) = \Pr[\mathcal{A} \text{ wins}].$$

In [10], Galbraith and Mao proved that invisibility is equivalent to anonymity if a DCS scheme has the following property: a randomly chosen signing key would produce a randomly distributed signature for any message. The result of Galbraith and Mao can be easily extended to the ID-based setting. We omit the details here and readers can refer to [10] for the detailed proof.

3.4.4. Unfoolability: security for verifier

Unfoolability means an adversary cannot fool a verifier in the *Confirm* or *Disavowal* protocol, even if the adversary knows all the secret keys. We define unfoolability via the following game between a challenger $\mathcal{V}$ and an adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$.

- Setup and Training Phases: same as in the unforgeability game.
- Output Phase: The adversary $\mathcal{A}_0$ outputs a target message m^*, a DCS signature σ'^*, a signer's identity ID_S^*, a confirmer's identity ID_C^*, and three auxiliary outputs τ_1, τ_2, τ_3. The ConfirmedSign, Confirm and Disavowal protocols are then executed by $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$, respectively, that is,
  - $(b_1, \sigma') \leftarrow \mathcal{CS}'_{(\mathcal{A}_1(\tau_1),\mathcal{V})}(m^*, \mathrm{ID}_S^*, \mathrm{ID}_C^*)$,
  - $b_2 \leftarrow \mathcal{CF}'_{(\mathcal{A}_2(\tau_2),\mathcal{V})}(m^*, \sigma'^*, \mathrm{ID}_S^*, \mathrm{ID}_C^*)$,
  - $b_3 \leftarrow \mathcal{DV}'_{(\mathcal{A}_3(\tau_3),\mathcal{V})}(m^*, \sigma'^*, \mathrm{ID}_S^*, \mathrm{ID}_C^*)$.

  Define Boolean variables

$$\alpha_1 = (b_1 \wedge \mathcal{VF}'(mpk, \mathrm{ID}_S^*, m^*, \mathcal{ES}'(m^*, \sigma', SK_{\mathrm{ID}_C^*}, \mathrm{ID}_S^*)) = 0),$$
$$\alpha_2 = (b_2 \wedge \mathcal{VF}'(mpk, \mathrm{ID}_S^*, m^*, \mathcal{ES}'(m^*, \sigma'^*, SK_{\mathrm{ID}_C^*}, \mathrm{ID}_S^*)) = 0),$$
$$\alpha_3 = (b_3 \wedge \mathcal{VF}'(mpk, \mathrm{ID}_S^*, m^*, \mathcal{ES}'(m^*, \sigma'^*, SK_{\mathrm{ID}_C^*}, \mathrm{ID}_S^*)) = 1).$$

We say the adversary $\mathcal{A}$ wins the game if $\alpha_1 \vee \alpha_2 \vee \alpha_3 = 1$. The advantage of the adversary is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{unfool}}(k) = \Pr[\mathcal{A} \text{ wins}].$$

4. Generic construction of anonymous IBDCS

In this section, we present a generic construction of anonymous IBDCS, which consists of a tuple of algorithms $(\mathcal{ST}', \mathcal{ET}', \mathcal{SG}', \mathcal{VF}', \mathcal{CS}', \mathcal{CF}', \mathcal{DV}', \mathcal{ES}')$ as defined in Section 3.3, based on several building blocks reviewed above, namely an anonymous IBS scheme $(\mathcal{ST}, \mathcal{ET}, \mathcal{SG}, \mathcal{VF})$, an IDOWKE scheme $(Setup, Extract, KI, KR)$, and a trapdoor hash function (KG, H_{HK}). Our generic construction will simply treat each building block as a black-box module by invoking its functions without describing the actual steps of those functions.

- $\mathcal{ST}'(1^k)$:
  (1) Run the setup algorithm of the IBS scheme to obtain $(param_1, (mpk_1, msk_1)) \leftarrow \mathcal{ST}(1^k)$.
  (2) Run the setup algorithm of the IDOWKE scheme to obtain $(param_2, (mpk_2, msk_2)) \leftarrow Setup(1^k)$.
  (3) Run the key generation algorithm of the trapdoor hash function to generate $(HK, TK) \leftarrow KG(1^k)$. For simplicity, in the rest of the paper, we will use H to denote H_{HK}.
  (4) Return $param = (param_1, param_2, HK)$, $mpk = (mpk_1, mpk_2)$ and $msk = (msk_1, msk_2)$.
- $\mathcal{ET}'(msk, \mathrm{ID})$:
  (1) Run the key extraction algorithm of the IBS scheme to obtain $SK_{\mathrm{ID}}^1 \leftarrow \mathcal{ET}(msk_1, \mathrm{ID})$.

(2) Run the key extraction algorithm of the IDOWKE scheme to obtain $SK_{\mathrm{ID}}^2 \leftarrow Extract(msk_2, \mathrm{ID})$.
(3) Return $SK_{\mathrm{ID}} = (SK_{\mathrm{ID}}^1, SK_{\mathrm{ID}}^2)$.

- $\mathcal{SG}'(SK_{\mathrm{ID}}, M)$:
  (1) Randomly select R for the trapdoor hash function H.
  (2) Run the IBS signing algorithm to generate $\sigma'' \leftarrow \mathcal{SG}(SK_{\mathrm{ID}}^1, H(M;R))$.
  (3) Return the signature as $\sigma = (\sigma'', R)$.
- $\mathcal{VF}'(mpk, \mathrm{ID}_S, M, \sigma)$:
  (1) Run the IBS verification algorithm $b \leftarrow \mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'')$.
  (2) Return b.
- $\mathcal{CS}'_{(S,V)}(M, \mathrm{ID}_S, \mathrm{ID}_C)$:
  (1) The signer S runs the IBOWKE protocol to generate $(M_{SC}, R) \leftarrow KI(\mathrm{ID}_C)$.
  (2) The signer S then runs the IBS signing algorithm to generate $\sigma'' \leftarrow \mathcal{SG}(SK_{\mathrm{ID}_S}^1, H(M;R)))$ and sets the DCS $\sigma' = (M_{SC}, \sigma'')$.
  (3) The signer finally uses a zero-knowledge proof to prove to V that

$$(M_{SC}, R) = KI(\mathrm{ID}_C) \wedge \mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 1.$$

- $\mathcal{CF}'_{(C,V)}(M, \sigma', \mathrm{ID}_S, \mathrm{ID}_C))$:
  (1) The confirmer C first parses the DCS σ' into (M_{SC}, σ''), and then calculates $R = KR(SK_{\mathrm{ID}_C}^2, M_{SC})$.
  (2) C then checks whether $\mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 1$.
  (3) If the check fails, it aborts the protocol. Otherwise, C performs a zero-knowledge proof to V that

$$R = KR(SK_{\mathrm{ID}_C}^2, M_{SC}) \wedge \mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 1.$$

- $\mathcal{DV}'_{(C,V)}(M, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$:
  (1) The confirmer C first parses the DCS σ' into (M_{SC}, σ''), and then calculates $R = KR(SK_{\mathrm{ID}_C}^2, M_{SC})$.
  (2) C then checks whether $\mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 0$.
  (3) If the check fails, it aborts the protocol. Otherwise, C performs a zero-knowledge proof to V that

$$R = KR(SK_{\mathrm{ID}_C}^2, M_{SC}) \wedge \mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 0.$$

- $\mathcal{ES}'(M, \sigma', SK_{\mathrm{ID}_C}, \mathrm{ID}_S)$:
  (1) The confirmer C parses the DCS σ' into (M_{SC}, σ''), and then calculates $R = KR(SK_{\mathrm{ID}_C}^2, M_{SC})$.
  (2) C then checks whether $\mathcal{VF}(mpk_1, \mathrm{ID}_S, H(M;R), \sigma'') = 1$.
  (3) If the check fails, it outputs $\perp$. Otherwise, C returns $\sigma = (\sigma'', R)$ as the signature of ID_S on message M.

4.1. Undeniable signature scheme

It is easy to see that the above IBDCS scheme also implies a *convertible* ID-based undeniable signature scheme when the signer puts him/herself as the designated confirmer. Specifically, an undeniable signature can be generated by modifying the $\mathcal{CS}'$ algorithm as follows.

(1) The signer S runs the IBOWKE protocol to generate $(M_{SS}, R) \leftarrow KI(\mathrm{ID}_S)$.
(2) S then runs the IBS signing algorithm to generate $\sigma'' \leftarrow \mathcal{SG}(SK_{\mathrm{ID}_S}^1, H(M;R)))$ and sets the undeniable signature as $\sigma' = (M_{SC}, \sigma'')$.

By doing a similar modification on the $\mathcal{CF}', \mathcal{DV}', \mathcal{ES}'$ algorithms, we can also obtain the corresponding confirm, disavowal, and extract signature algorithms of the undeniable signature scheme.

5. Security analysis

We prove that the proposed generic IBDCS scheme is secure if all the underlying building blocks are secure. Specifically, we prove that our IBDCS scheme can satisfy all the security properties given in Section 3.4.

Theorem 5.1: *The above IBDCS scheme is UF-CMIDA secure, given that the underlying IBS is UF-CMIDA secure, and the trapdoor hash function is collision resistant.*

Proof: The proof is by contradiction. Suppose there exists an adversary $\mathcal{F}_1$ that can break the UF-CMIDA security (denote the corresponding game as $Game_1$) of the IBDCS scheme with advantage $\mathrm{Adv}_{\mathcal{F}_1}^{\mathrm{UF-CMIDA}}(k)$, we construct another algorithm $\mathcal{F}_2$ that can break the UF-CMIDA security (denote the corresponding game as $Game_2$) of the IBS scheme. ■

Upon receiving the system parameters $param_1$ and mpk_1 in $Game_2$, $\mathcal{F}_2$ generates a key pair (HK, TK) of the trapdoor hash function and $(param_2, (mpk_2, msk_2))$ of the IDOWKE scheme. Let H denote the trapdoor hash function defined by HK. $\mathcal{F}_2$ then forwards $(param_1, param_2), H, (mpk_1, mpk_2)$ to $\mathcal{F}_1$.

$\mathcal{F}_2$ makes a random guess that user i is the target user that $\mathcal{F}_1$ will forge a DCS signature. $\mathcal{F}_2$ makes Extract queries to its own oracle to obtain $SK^1_{\mathrm{ID}_j}$ for any $j \neq i$, and also generates $SK^2_{\mathrm{ID}_\ell}$ for each user ℓ. $\mathcal{F}_2$ answers $\mathcal{F}_1$'s queries as follows:

For each Key Extraction query made by $\mathcal{F}_1$, $\mathcal{F}_2$ can answer the query directly except that if $\mathcal{F}_1$ makes a Key Extraction query for ID_i, $\mathcal{F}_2$ aborts the game.

For each ConfirmSign query made by $\mathcal{F}_1$ on input $(m, \mathrm{ID}_S, \mathrm{ID}_C)$, if $\mathrm{ID}_S \neq \mathrm{ID}_i$, then $\mathcal{F}_2$ can simulate the ConfirmedSign protocol perfectly using the knowledge of $(SK^1_{\mathrm{ID}_S}, SK^2_{\mathrm{ID}_S})$. If $\mathrm{ID}_S = \mathrm{ID}_i$, then $\mathcal{F}_2$ first runs $KI(\mathrm{ID}_C)$ to generate (M_{SC}, R), and then makes a signing query to its own oracle for ID_i and $H(M; R)$. After obtaining the signature σ'', $\mathcal{F}_2$ completes the zero-knowledge proof based on the knowledge of the randomness used in running $KI(\mathrm{ID}_C)$. The DSC is $\sigma' = (M_{SC}, \sigma'')$.

For the Confirm, Disavowal, and ExtractSignature queries, $\mathcal{F}_2$ can answer them perfectly since $\mathcal{F}_2$ has the knowledge of $SK^2_{\mathrm{ID}_\ell}$ for each ID_ℓ.

Finally, $\mathcal{F}_1$ outputs a forgery (m^*, σ'^*) for signer ID^*_S and confirmer ID^*_C. If $\mathrm{ID}^*_S \neq \mathrm{ID}_i$, $\mathcal{F}_2$ aborts the game. Otherwise, $\mathcal{F}_2$ runs $\mathcal{ES}'(m^*, \sigma'^*, SK_{\mathrm{ID}^*_C}, \mathrm{ID}^*_S)$ to extract a signature (R^*, σ''^*). If the extraction fails, $\mathcal{F}_2$ aborts the game; otherwise, $\mathcal{F}_2$ checks if it has ever made a signing query on $(\mathrm{ID}^*_S, H(m^*; R^*))$. If no such query has been made, $\mathcal{F}_2$ outputs $(H(m^*; R^*), \sigma''^*)$ as its forgery for the IBS scheme; otherwise, $\mathcal{F}_2$ aborts the game.

Analysis. Let $\mathbf{E}_1$ be the event that $\mathrm{ID}^*_S = \mathrm{ID}_i$. Since $\mathcal{F}_2$ chooses i randomly, we have

$$\Pr[\mathbf{E}_1] = \frac{1}{n},$$

where n denotes the number of users in the system. Under the condition of $\mathbf{E}_1$, we can see that the game simulated by $\mathcal{F}_2$ is perfect.

Let $\mathbf{E}_2$ be the event that signature σ'^* is a valid DCS for message m^*, which means the confirmer can successfully extract the full signature σ^* from σ'^*, and $\mathbf{E}_3$ the event that $(\mathrm{ID}^*_S, H(m^*; R^*))$ has appeared in the Signing queries made by $\mathcal{F}_2$. Then we have

$$\begin{aligned}
\mathrm{Adv}_{\mathcal{F}_2}^{\mathrm{uf-cmida}}(k) &= \Pr[\mathbf{E}_1 \wedge \mathbf{E}_2 \wedge \bar{\mathbf{E}_3}] \\
&= \Pr[\mathbf{E}_1]\Pr[\mathbf{E}_2 \wedge \bar{\mathbf{E}_3}|\mathbf{E}_1] \\
&= \Pr[\mathbf{E}_1](1 - \Pr[\bar{\mathbf{E}_2} \vee \mathbf{E}_3|\mathbf{E}_1]) \\
&\geq \Pr[\mathbf{E}_1](1 - (\Pr[\bar{\mathbf{E}_2}|\mathbf{E}_1] + \Pr[\mathbf{E}_3|\mathbf{E}_1])) \\
&= \Pr[\mathbf{E}_1](\Pr[\mathbf{E}_2|\mathbf{E}_1] - \Pr[\mathbf{E}_3|\mathbf{E}_1])
\end{aligned}$$

According to the security model, $\mathcal{F}_1$ cannot ask the ConfirmedSign Query for (ID_S^*, m^*). Hence, if event E_3 occurs, there must exist another $(\mathrm{ID}_S^*, m', R')$ which has appeared in a ConfirmedSign Query and $H(m'; R') = H(m^*; R^*)$. Since the trapdoor hash function is collision resistant, we have

$$\Pr[\mathbf{E}_3|\mathbf{E}_1] \leq \mathrm{Adv}_{\mathcal{A}}^{CR}(k),$$

where $\mathrm{Adv}_{\mathcal{A}}^{CR}(k)$ denotes the probability for a collision finder $\mathcal{A}$ to break the collision resistant property of the trapdoor hash function.

Combing all together, we have

$$\mathrm{Adv}_{\mathcal{F}_1}^{\mathrm{UF-CMIDA}}(k) \leq n\mathrm{Adv}_{\mathcal{F}_2}^{\mathrm{uf-cmida}}(k) + \mathrm{Adv}_{\mathcal{A}}^{CR}(k).$$

Therefore, if the underlying IBS is unforgeable (i.e. $\mathrm{Adv}_{\mathcal{F}_2}^{\mathrm{uf-cmida}}(k)$ is negligible), and the hash function is collision resistant (i.e. $\mathrm{Adv}_{\mathcal{A}}^{CR}(k)$ is negligible), then our IBDCS is also unforgeable (i.e. $\mathrm{Adv}_{\mathcal{F}_1}^{\mathrm{UF-CMIDA}}(k)$ is negligible).

Theorem 5.2: *The above IBDCS scheme is SA-CMIDA secure, given that the underlying IBS scheme is SA-CMIDA secure, the IDOWKE scheme has key privacy, the Confirm and Disavowal protocols are zero-knowledge.*

Proof: We prove this theorem by a sequence of games {**Game i**}. Let Pr[**Game i**] denote the probability that the adversary $\mathcal{D}$ guesses the value of b correctly in Game i.

Game 0: This is the original anonymity game for IBDCS. The advantage $\mathcal{D}$ can win the game is:

$$\Pr[\textbf{Game 0}] - \frac{1}{2} = \mathrm{Adv}_{\mathcal{D}}^{\mathrm{SA-CMIDA}}(k).$$

Game 1: This game is the same as Game 0 except that when answering the Confirm or Disavowal query, the challenger uses the zero-knowledge simulator to do the proof if ID_C has not appeared in any Extract query. Since the Confirm and Disavowal protocols are zero-knowledge, we have

$$\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}] \leq \mathrm{Adv}_{\mathcal{D}_1}^{ZK}(k).$$

Game 2: This game is the same as Game 1 except that the challenge DCS σ'^* in the challenge phase is generated as follows: the simulator replaces the value of M_{SC} in σ'^* by a random IDOWKE message $\hat{M}_{SC} \leftarrow KI(\mathrm{ID}_C)$. ■

Lemma 1: *Game 1 and Game 2 are indistinguishable to $\mathcal{D}$ if the IDOWKE protocol has key privacy.*

Proof: By contradiction, suppose there exists an adversary $\mathcal{D}$ which can distinguish Game 1 and Game 2, we construct another adversary $\mathcal{D}_2$ that can break the key privacy of the IDOWKE protocol.

After receiving *param* and *mpk* for the IDOWKE protocol, $\mathcal{D}_2$ simulates the game for $\mathcal{D}$ by generating the master keys for the IBS scheme, (HK, TK) for the trapdoor hash function, and setting $param_2 = param, mpk_2 = mpk$.

$\mathcal{D}_2$ answers the Extract queries by querying its own oracle, and ConfirmedSign queries as usual. For each Confirm, Disavowal, or ExtractSignature query on input $(m, \sigma', \mathrm{ID}_S, \mathrm{ID}_C)$, $\mathcal{D}_2$ parses $\sigma' = (M_{SC}, \sigma'')$, makes a Send query with input (M_{SC}, ID_C) to its challenger, and then makes a Reveal query to obtain $R = KR(SK_{\mathrm{ID}_C}^2, M_{SC})$. In order to perform the zero-knowledge proof in the Confirm

or Disavowal protocol, $\mathcal{D}_2$ uses the zero-knowledge simulator to do the proof if ID_C has not appeared in any Extract query.

After $\mathcal{D}$ outputs $\mathrm{ID}^*_{S_0}, \mathrm{ID}^*_{S_1}, \mathrm{ID}^*_C$ and m^* in the Key Selection Phase, $\mathcal{D}_2$ outputs ID^*_C as the challenge identity and receives the challenge (M^*, K^*) for the IDOWKE protocol. $\mathcal{D}_2$ randomly chooses $\hat{b} \leftarrow \{0, 1\}$ and computes the challenge DCS for message m^* as

$$\sigma'^* = (M^*, \mathcal{SG}(SK^1_{\mathrm{ID}^*_{S_{\hat{b}}}}, H(m^*; K^*))).$$

$\mathcal{D}_2$ then continues to answer the queries made by $\mathcal{D}$ as before. Finally, if $\mathcal{D}$ guesses $\hat{b}$ correctly, $\mathcal{D}_2$ outputs 1; otherwise, $\mathcal{D}_2$ outputs 0.

If K^* is the real key corresponding to M^* (i.e. $b = 1$), then $\mathcal{D}$ is in Game 0; otherwise, if K^* is a random key unrelated to M^* (i.e. $b = 0$), then $\mathcal{D}$ is in Game 1. Therefore, we have

$$\begin{aligned}\mathrm{Adv}^{KP}_{\mathcal{D}_2}(k) &= \Pr[b = 1](\Pr[\mathcal{D}_2 \to 1 | b = 1]) + \Pr[b = 0](\Pr[\mathcal{D}_2 \to 0 | b = 0]) - \frac{1}{2} \\ &= \frac{1}{2}\Pr[\mathbf{Game}_1] + \frac{1}{2}(1 - \Pr[\mathbf{Game}_2]) - \frac{1}{2} \\ &= \frac{1}{2}(\Pr[\mathbf{Game}_1] - \Pr[\mathbf{Game}_2]).\end{aligned}$$

■

Lemma 2: *$\mathcal{D}$ has a negligible advantage in Game 2.*

Proof: If $\mathcal{D}$ has a non-negligible advantage in Game 2, we can construct another adversary $\mathcal{D}_3$ to break the anonymity of the IBS scheme. ■

After receiving *param* and *mpk* for the IBS, $\mathcal{D}_3$ sets $param_1 = param, mpk_1 = mpk$ and generates (HK, TK) for the trapdoor hash function and $(param_2, mpk_2, msk_2)$ for the IDOWKE protocol. $\mathcal{D}_3$ answers Extract queries by using its own key extraction oracle. To answer the ConfirmedSign queries, $\mathcal{D}_3$ makes signing queries to its own oracle if the corresponding user secret key is unknown. $\mathcal{D}_3$ answers the Confirm, Disavowal and ExtractSignature queries based on the knowledge of msk_2.

When $\mathcal{D}$ outputs $\mathrm{ID}^*_{S_0}, \mathrm{ID}^*_{S_1}, \mathrm{ID}^*_C$ and m^* in the Key Selection Phase, $\mathcal{D}_3$ outputs $\mathrm{ID}^*_{S_0}, \mathrm{ID}^*_{S_1}$ as the challenge identities in the IBS anonymity game. After receiving the challenge signature σ^*, $\mathcal{D}_3$ generates $M^*_{SC} \leftarrow KI(\mathrm{ID}^*_C)$ and outputs $\sigma'^* = (M^*_{SC}, \sigma^*)$ as the challenge DCS for $\mathcal{D}$.

$\mathcal{D}_3$ then continues to simulate the game as before, and outputs whatever $\mathcal{D}$ outputs.

Since in the IBS anonymity game, σ^* is a valid signature for a random element h selected from the range of the trapdoor hash function H, and due to the inversion property of H, there exists an R such that $H(m^*; R) = h$. Moreover, R is uniformly distributed when h is randomly chosen. Therefore, $\mathcal{D}_3$ simulates Game 2 perfectly and we have

$$\Pr[\mathbf{Game}_2] = \mathrm{Adv}^{\mathrm{sa-cmida}}_{\mathcal{D}_3}(k) + \tfrac{1}{2}.$$

Based on Lemmas 1 and 2, we have

$$\mathrm{Adv}^{\mathrm{SA-CMIDA}}_{\mathcal{D}}(k) \leq \mathrm{Adv}^{ZK}_{\mathcal{D}_1}(k) + 2\mathrm{Adv}^{KP}_{\mathcal{D}_2}(k) + \mathrm{Adv}^{\mathrm{sa-cmida}}_{\mathcal{D}_3}(k).$$

Therefore, if the underlying zero-knowledge proofs are secure (i.e. $\mathrm{Adv}^{ZK}_{\mathcal{D}_1}(k)$ is negligible), the underlying IDOWKE scheme is secure (i.e. $\mathrm{Adv}^{KP}_{\mathcal{D}_2}(k)$ is negligible), and the underlying IBS scheme is anonymous (i.e. $\mathrm{Adv}^{\mathrm{sa-cmida}}_{\mathcal{D}_3}(k)$ is negligible), then our IBDCS is also anonymous (i.e. $\mathrm{Adv}^{\mathrm{SA-CMIDA}}_{\mathcal{D}}(k)$ is negligible).

Invisibility proof. As mentioned in Section 3.4.3, anonymity and invisibility of an IBDCS can be proven equivalent by following the result of Galbraith and Mao [10]. Therefore, we omit the detailed proof for invisibility here.

Theorem 5.3: *The generic IBDCS scheme satisfies unfoolability under adaptive chosen message and ID attacks, given that the zero-knowledge proofs used in the ConfirmedSign, Confirm, and Disavowal protocols satisfy soundness.*

Proof: The proof is by contradiction. Given an adversary $\mathcal{A}_1$ that can break the unfoolability of the IBDCS scheme, we construct another algorithm $\mathcal{A}_2$ that can break the soundness of the zero-knowledge proof used in ConfirmedSign, Confirm, or Disavowal protocol.

$\mathcal{A}_2$ generates all the system parameters and master keys honestly, and answers all the oracle queries made by $\mathcal{A}_1$ in the Training Phase by following the algorithms honestly. After receiving $(m^*, \sigma'^*, \mathrm{ID}_S^*, \mathrm{ID}_C^*)$ from $\mathcal{A}_1$ in the output phase, $\mathcal{A}_2$ selects a random $i \in \{1, 2, 3\}$. $\mathcal{A}_2$ runs the zero-knowledge proof in the ConfirmedSign/Confirm/Disavowal protocol with an honest verifier V for $i = 1/2/3$, respectively. $\mathcal{A}_2$ simply relays all the messages sent between $\mathcal{A}_1(\tau_i)$ and V . It is then easy to see that if $\mathcal{A}_1$ can win the unfoolability game with advantage ϵ, then $\mathcal{A}_2$ can break the soundness of the zero-knowledge proof used in ConfirmedSign, Confirm, or Disavowal protocol with advantage at least $\epsilon/3$. ■

6. Instantiation of the generic construction

Since our generic construction treats the underlying building blocks (namely, IDOWKE, IBS, and trapdoor hash function) as black boxes, we can use any concrete schemes to instantiate these building blocks in order to implement the proposed anonymous IBDCS scheme. Hence, our generic construction provides an engineering way to build concrete IBDCS schemes. However, we should note that the 'interoperability' among those building blocks must be considered in real implementations. In particular, it is desirable to choose building blocks that can share the same system parameters.

There are many concrete examples of those building blocks that can suit our need. As an example, to instantiate the IDOWKE, we can use the protocol proposed by Gorantla *et al.* [14] which is based on bilinear groups. We can then choose an anonymous IBS scheme that can share the same set of system parameters with the IDOWKE scheme, such as the pairing-based IBS scheme proposed by Paterson and Schuldt [23]. For the trapdoor hash function, we can use the Discrete Log-based trapdoor hash function proposed by Krawczyk and Rabin [19] to match the chosen IDOWKE and IBS schemes.

On the other hand, the instantiation and implementation of the zero-knowledge proof is more challenging and requires substantial work. Although in theory we can always build zero-knowledge proof systems for our ConfirmedSign, Confirm, and Disavowal protocols based on the fact that generic zero-knowledge proof exists for any NP language [12], the resulting protocols will become very inefficient due to the complex NP reduction. Hence, for the sake of efficiency, it is desirable to avoid the generic zero-knowledge proofs in real implementations. One potential candidate for our scheme is the Groth–Sahai (non-interactive) zero-knowledge proof system [15], which can efficiently perform zero-knowledge proofs for a variety of equations over bilinear groups. The Groth–Sahai system is suitable for our purpose since it can be used to prove the knowledge of a message-signature pair without revealing them to the verifier, which is required in our ConfirmedSign, Confirm, and Disavowal protocols. However, the Groth–Sahai systems requires the underlying signature scheme to satisfy the property of structure-preserving, which means the verification keys, signatures, and messages are all elements in a bilinear group, and the verification equation is a conjunction of pairing-product equations [2]. Recently, several structure-preserving digital signature schemes have been proposed [1–3]. By constructing an efficient anonymous and structure-preserving identity-based signature scheme, we are able to efficiently instantiate the zero-knowledge proof and hence the entire

anonymous IBDCS scheme. We leave the instantiation and implementation of our generic IBDCS scheme as our future work.

7. Applications

In this section, we present some applications of the proposed anonymous IBDCS scheme.

Fair Contract Signing. The first application is a fair electronic contract signing protocol briefly described in the introduction. Suppose that Alice and Bob wants to remotely sign an electronic contract. However, if one party, say Alice, has first obtained the signature of Bob, then Alice may refuse to send her signature to Bob but use Bob's signature to bargain with another competitor of Bob. In order to avoid such a situation, we can use the proposed IBDCS scheme as follows:

(1) Alice first runs the ConfirmedSign protocol to generate an IBDCS σ'_A on the contract and proves to Bob that her valid signature σ_A on the contract is encapsulated in σ'_A. Let Carol be the semi-trusted confirmer who can execute the ExtractSignature algorithm to recover σ_A from σ'_A.
(2) After verifying that σ'_A is a valid IBDCS in the ConfirmedSign protocol, Bob sends his valid σ_B on the contract and the received IBDCS σ'_A to the confirmer Carol.
(3) Carol first verifies that σ_B is a valid signature of Bob on the contract. Then Carol runs the ExtractSignature algorithm to obtain σ_A from σ'_A. Finally, Carol send σ_B to Alice, and σ_A to Bob.

In the above protocol, we can ensure the fairness between Alice and Bob. Although Bob first receives an IBDCS σ'_A from Alice, he cannot use it to convince another party that Alice has signed the contract due to the invisibility of the signature. In fact, due to the anonymity property of our IBDCS scheme, Bob even cannot prove to another party σ'_A is from Alice. On the other hand, Bob can be ensured that σ'_A indeed contains a valid signature of Alice due to the unfoolability of the IBDCS. Therefore, he can obtain from the confirmer Carol a valid signature of Alice on the contract if he honestly signs the contract and sends his signature to Carol. Moreover, since our proposed scheme is ID-based, no digital certificate is needed for any party.

E-Cheque. The second application of our proposed IBDCS scheme is an electronic cheque protocol. Suppose that Alice is looking for a freelance worker to do a job (e.g. writing a computer program), and Bob is willing to take the job. Alice may wish to give Bob a payment cheque after the job is finished, while Bob is concerned that Alice may not do the payment after receiving the product. To solve the problem, we can also employ the proposed IBDCS scheme as follows:

(1) Alice first runs the ConfirmedSign protocol to generate an IBDCS σ'_A on the cheque and proves to Bob that σ'_A contains a valid signature σ_A on the cheque.
(2) After verifying that σ'_A is a valid IBDCS in the ConfirmedSign protocol, Bob starts to do the job. After the job is finished, Bob sends the final product to Alice. Meanwhile, Bob also sends the IBDCS σ'_A and a proof that he has finished the job to the confirmer Carol.
(3) After verifying that Bob has completed the job, Carol runs the ExtractSignature algorithm to obtain σ_A from σ'_A. Finally, Carol sends σ_A to Bob.

In the above protocol, we can also ensure the fairness between the payer and the payee since Bob can receive the cheque only after he finishes the job, while Alice cannot refuse to do the payment after the job is finished since she needs to first commit her signature on the cheque in the IBDCS and Bob can verify it in the ConfirmedSign protocol before starting to do the job.

8. Conclusion and future work

In this paper, we proposed a new generic construction of Anonymous IBDCS. Different from the previous widely used Sign-then-Encrypt approach which relies on public-key encryption to hide a

signature from a verifier, our construction first embeds some secret information, which can only be derived by the designated confirmer, into the message being signed, and then uses an anonymous signature scheme to generate the final signature, which is only verifiable and convertible by the designated confirmer. Interestingly, our construction also directly leads to a new way to build convertible undeniable signatures. We proved that our proposed scheme is secure and presented several applications of it. We also provided some directions on the efficient instantiation of the proposed scheme, which is left as our future work.

Acknowledgments

We thank the anonymous reviewers for their invaluable comments and suggestions that have greatly helped us improve our work.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

[1] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo, *Constant-size structure-preserving signatures: Generic constructions and simple assumptions*, ASIACRYPT, 2012, pp. 4–24.
[2] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo, *Structure-preserving signatures and commitments to group elements*, CRYPTO, 2010, pp. 209–236
[3] M. Abe, J. Groth, M. Ohkubo, and M. Tibouchi, *Structure-preserving signatures from Type II pairings*, CRYPTO, 2014, pp. 390–407.
[4] J. Camenisch and M. Michels, *Confirmer signature schemes secure against adaptive adversaries*, EUROCRYPT, 2000, pp. 243–258.
[5] J. Camenisch and V. Shoup, *Practical verifiable encryption and decryption of discrete logarithms*, CRYPTO, 2003, pp. 126–144.
[6] D. Chaum, *Zero-knowledge undeniable signatures*, EUROCRYPT, 1990, pp. 458–464.
[7] D. Chaum, *Designated confirmer signatures*, EUROCRYPT, 1994, pp. 86–91.
[8] D. Chaum and H.V. Antwerpen, *Undeniable signatures*, CRYPTO, 1989, pp. 212–216.
[9] M. Fischlin, *Anonymous signatures made easy*, Public Key Cryptography, 2007, pp. 31–42.
[10] S.D. Galbraith and W. Mao, *Invisibility and anonymity of undeniable and confirmer signatures*, Topics in Cryptology – CT-RSA, 2003, pp. 80–97.
[11] C. Gentry, D. Molnar, and Z. Ramzan, *Efficient designated confirmer signatures without random oracles or general zero-knowledge proofs*, ASIACRYPT, 2005, pp. 662–681.
[12] O. Goldreich, S. Micali, and A. Wigderson, *Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems*, J. ACM 38 (1991), pp. 691–729.
[13] S. Goldwasser and E. Waisbard, *Transformation of digital signature schemes into designated confirmer signature schemes*, TCC, 2004, pp. 77–100
[14] M.C. Gorantla, C. Boyd, and J.M.G. Nieto, *ID-based one-pass authenticated key establishment*, Australasian Information Security Conference (AISC), 2008.
[15] J. Groth and A. Sahai, *Efficient non-interactive proof systems for bilinear groups*, EUROCRYPT, 2008, pp. 415–432.
[16] Q. Huang, D.S. Wong, and W. Susilo, *A new construction of designated confirmer signature and its application to optimistic fair exchange – (extended abstract)*, Pairing, 2010, pp. 41–61.
[17] Q. Huang, D.S. Wong, and W. Susilo, *The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles*, Public Key Cryptography, 2012, pp. 120–137.
[18] Q. Huang, G. Yang, D.S. Wong, and W. Susilo, *Ambiguous optimistic fair exchange*, Advances in Cryptology – ASIACRYPT 2008, pp. 74–89.
[19] H. Krawczyk and T. Rabin, *Chameleon signatures*, Proceedings of the Network and Distributed System Security Symposium NDSS, 2000.
[20] M. Michels and M. Stadler, *Generic constructions for secure and efficient confirmer signature schemes*, Advances in Cryptology EUROCRYPT'98, Springer, 1998, pp. 406–421.
[21] J. Monnerat and S. Vaudenay, *Chaum's designated confirmer signature revisited*, ISC, 2005, pp. 164–178.
[22] T. Okamoto, *Designated confirmer signatures and public-key encryption are equivalent*, CRYPTO, 1994, pp. 61–74.

[23] K.G. Paterson and J.C.N. Schuldt, *Efficient identity-based signatures secure in the standard model*, Australasian Conference on Information Security and Privacy ACISP, 2006, pp. 207–222.
[24] A. Shamir, *Identity-based cryptosystems and signature schemes*, CRYPTO, 1984, pp. 47–53.
[25] G. Wang, J. Baek, D.S. Wong, and F. Bao, *On the generic and efficient constructions of secure designated confirmer signatures*, Public Key Cryptography, 2007, pp. 43–60.
[26] F. Xia, G. Wang, and R. Xue, *On the invisibility of designated confirmer signatures*, ASIACCS, 2011, pp. 268–276
[27] G. Yang, D.S. Wong, X. Deng, and H. Wang, *Anonymous signature schemes*, Public Key Cryptography, 2006, pp. 347–363.
[28] F. Zhang, X. Chen, and B. Wei, *Efficient designated confirmer signature from bilinear pairings*, ASIACCS, 2008, pp. 363–368.
[29] R. Zhang and H. Imai, *Strong anonymous signatures*, Inscrypt, 2008, pp. 60–71.