Privacy-enhanced Designated Confirmer Signature without Random Oracles

Shengke Zeng[1,2] and Hu Xiong[1]

(Corresponding author: Shengke Zeng)

School of Computer Science and Engineering, University of Electronic Science and Technology of China[1]
Chengdu, 611731, P. R. China

School of Mathematics and Computer Engineering, Xihua University, Chengdu, 610039, P. R. China[2]

(Email: zengshengke@gmail.com)

(Received Nov. 9, 2012; revised and accepted Jan. 18, 2013)

Abstract

As an extension of digital signature, designated confirmer signature (DCS) efficiently realizes the privacy protection of the signer. In a DCS scheme, the validity of the signature must be confirmed by the signer or a semi-trusted third party, called *confirmer*. Since the DCS signature is generated by encrypting a standard signature with the designated confirmer's public key, only the confirmer can disavow invalid signatures. Moreover, the confirmer always can further convert a DCS into an ordinary signature by decrypting the DCS such that it is publicly verifiable. This property is necessary in some cases. However, from the view of the signer, the privacy is leaked if the DCS is converted into an ordinary signature by the confirmer. In this work, we propose a new DCS scheme without the random oracles. Both the signer and confirmer in this new construction can confirm a valid DCS and disavow an invalid signature. Furthermore, the authority of the confirmer is limited. He is designated by the signer to verify the validity of a signature only when the signer is unavailable. However, the confirmer cannot convert a valid DCS into a standard signature. This new property of DCS is more favorable to the signer.

Keywords: Designated Confirmer Signature, Standard Model, Privacy-preserving Signature

1 Introduction

Digital signature is a publicly verifiable scheme that any verifier can be convinced the integrity and authenticity of a message sent by the signer. Therefore, the validity of a signature can be shown to anybody. In some cases, the signer does not want his signature to be verified or transferred by anyone. Hence he must control the public verifiability of the signature. Chaum and van Antwerpen [6] solved this problem by introducing *undeniable signature*. In an undeniable signature scheme, the signature cannot be directly verified by the receiver. The receiver needs the help of the signer to verify the validity of the signature and the receiver cannot show the validity of the signature to others. Hence, the signer can choose the qualified verifiers (e.g. prepaid customers) to verify his signature. In order to alleviate the burden of the signer or when the signer is unavailable, we need a semi-trusted third party to assist the signer on the verification. To achieve this goal, Chaum [4] introduced the notion of *designated confirmer signature* (DCS). In a DCS scheme, the signer designates a semi-trusted third party, called *confirmer* to confirm or disavow a signature on behalf of the signer. Moreover, the confirmer in the DCS scheme can convert a designated confirmer signature into an ordinary signature such that it is publicly verifiable. However, the confirmer cannot forge the signer's signature.

1.1 Related Work

After the introduction of DCS [4], many constructions of DCS have been proposed. Okamoto [13] constructed the first DCS scheme and proved that a DCS scheme is equivalent to the public key encryption. However, the insecurity of [13] was pointed by Michels and Stadler [12] that the confirmer can forge the signer's signature. Gentry et al. [8] presented a DCS scheme by using a commitment scheme. However, Wang et al. [14] identified that two security flaws exist in Gentry's scheme: the confirmer and the signer can collude together to convince a verifier an invalid signature and the validity of DCS can be checked without the confirmer's help. Wei et al. [16] proposed a new notion of society-oriented designated confirmer signature based on threshold cryptography.

Most previous DCS schemes (including above proposals) follow the approach that the DCS signature is encrypted by using the confirmer's public key. Therefore, the signer does not have the capability to disavow an invalid signature. However, in many applications, it is necessary for the signer to have the same ability as the confirmer to disavow any invalid signatures. Galbraith and Mao [7] pointed out that DCS schemes should allow the signer to deny invalid signatures. Motivated by this observation, Huang et al. [10] proposed a DCS scheme to support the signer's disavowal. Their scheme is proven to be secure without the random oracles. Wang et al. [15] constructed a DCS scheme with unified verification. Compared to [10], Wang et al's scheme [15] is secure under the random oracles. However, [15] considered the verification in the concurrent execution environment. When transformed into concurrent zero-knowledge setting, the confirmation and disavowal protocols in [10] are less efficient in both computation and communication than [15].

1.2 Motivation and Contribution

Similar to [10] and [15], most of previous works produce the DCS signature by encrypting a standard signature with the designated confirmer's public key. Therefore, the confirmer in the DCS model can extract the ordinary signature produced by the signer. This property is necessary in some cases. However, the confirmer is a semi-trusted third party. If the confirmer converts the DCS signature into ordinary signature without signer's consent, the signer's privacy cannot be protected any more.

Let us consider the digitization system of healthcare. Such system provides comprehensive diagnosis and treatment for remote patients. The sensors monitor the patient and report the information to the panel doctors. For the validity of the data, the information should be authenticated by the patient. However, for the privacy of patient, the authenticated information cannot be verified by others. Therefore, the patient produces a DCS signature for the information and designates a semi-trusted principal to assist him on the verification when he is unavailable. Hence, the patient controls the verifiability by choosing the qualified verifiers (e.g. the panel doctors). However, if the designated confirmer is malicious, i.e. he threatens the patient for some economic disputes, he would convert the DCS signature into an original signature. Consequently, the privacy of the patient is leaked.

From the view of the signer, the confirmer designated by him is only to assist him to verify the validity of the signature for verifiers when he is unavailable. Hence the signer does not hope this confirmer has the ability to extract the signature. For this motivation, we modify the syntax of the traditional DCS slightly. That is the confirmer cannot convert the DCS signature. In this work, we propose a new efficient designated confirmer signature scheme which is secure without the random oracles. We require that in the new DCS model, both the signer and the confirmer can perform the confirmation protocol to confirm the validity of DCS signature and also can perform the disavowal protocol to deny an invalid DCS signature. Moreover, in order to protect the privacy of the signer, the confirmer cannot extract the ordinary signature from the DCS signature.

Organization. Section 2 introduces the preliminaries. Section 3 introduces the model of DCS scheme. Section 4 proposes the new DCS scheme. The performance and property comparison are presented in Section 5 and Section 6 gives the formal proofs for the security of our scheme. The last section is a conclusion.

2 Preliminary

2.1 Bilinear Pairings and BB signature

In this work, we make use of bilinear groups of composite order, which was introduced by Boneh, Goh and Nissim [2]. Let n be a composite with factorization $n = pq$. Then we have:

- G is a multiplicative cyclic group of order n. G_p is G's subgroup of order p and G_q is G's subgroup of order q.
- G_T is a multiplicative group of order n.
- $\hat{e} : G \times G \to G_T$ is an efficiently computable map with the following properties:
  - Bilinearity: $\forall u, v \in G,\ a, b \in \mathbb{Z}_n,\ \hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
  - Non-degeneracy: $\hat{e}(g, g)$ is the generator of G_T whenever g is the generator of G.
  - Computability: $\forall u, v \in G$, $\hat{e}(u, v)$ can be computed efficiently.
  - $G_{T,p}$ and $G_{T,q}$ are the G_T-subgroups of order p and q respectively.

Let us review the BB signature which was proposed by Boneh and Boyen [1]. It is a short signature scheme which is secure against an existential forgery under a weak chosen message attack provided that Strong Diffie-Hellman assumption is hard (see Definition 1). This signature scheme is secure in prime order group G_p and also can be adapted to composite order groups [3]. The BB signature scheme under the composite order bilinear group and symmetric pairing version is as follows:

- KeyGen. Given a group tuple $(n, G, G_T, \hat{e}, g)$, choose $x \in \mathbb{Z}_n$. The public key is $pk = g^x$ and the private key is $sk = x$.
- Signing. Given the private key $sk \in \mathbb{Z}_n$ and a message m, compute $\sigma = g^{\frac{1}{x+m}} \in G$.
- Verification. Given the public key pk, a message m and the signature σ, check that $\hat{e}(\sigma, pk \cdot g^m) = \hat{e}(g, g)$ holds or not. If it holds, accept the validity of (m, σ).

In fact, the original BB signature [1] is described in the asymmetric pairing setting, i.e. $\hat{e} : G_1 \times G_2 \to G_T$. It is trivial to use symmetric pairing by letting $G = G_1 = G_2$.

Theorem 1. *BB signature is unforgeable against weak chosen message attack if Strong Diffie-Hellman assumption holds.*

2.2 Complexity Assumptions

Definition 1 (Strong Diffie-Hellman Assumption in G_p). *The Strong Diffie-Hellman assmuption in G_p states that, given $\eta, \eta^x, \eta^{x^2}, \cdots, \eta^{x^t} \in G_p$ as input, there is no PPT adversary can output a pair $(c, \eta^{\frac{1}{x+c}})$, where $c \in \mathbb{Z}_p$.*

Definition 2 (Subgroup Decision Assumption). *Let $(n, G, G_T, \hat{e}, g)$ be the pairing parameters, where $n = pq$. The Subgroup Decision assumption states that it is hard to distinguish a random element in G from a random element in G_q.*

2.3 Zero-knowledge Proofs

Zero-knowledge proof system was first proposed by Goldwasser, Micali and Rackoff [9] which allows a prover P to convince a verifier V about the truth of the statement without revealing anything beyond the statement itself. For an $\mathcal{NP}$ language $\mathcal{L}$, any statement $x \in \mathcal{L}$ has a witness ω that allows one to efficiently verify the membership of x. All such pairs (x, ω) constitute a binary relation R. A pair of interactive algorithms $\langle P, V \rangle$ is called an interactive proof system for $\mathcal{L}$ if V is polynomial-time and the following two conditions hold:

- Completeness: For every $x \in \mathcal{L}$, there exists a string y (P's auxiliary input) such that for every $z \in \{0,1\}^*$ (V's auxiliary input), $\Pr[\langle P(y), V(z) \rangle(x) = 1] = 1$, where x is P and V's common input.
- Soundness: For every $x \notin \mathcal{L}$, every interactive machine P^* and every $y, z \in \{0,1\}^*$, $\Pr[\langle P^*(y), V(z) \rangle(x) = 1] = 0$.

Generally speaking, we say the interactive proof system $\langle P, V \rangle$ for language $\mathcal{L}$ is zero-knowledge if whatever can be efficiently computed after interacting with P on input $x \in \mathcal{L}$ can also be computed from x without any interaction. We say that $\langle P, V \rangle$ is zero-knowledge proof system if for every PPT interactive algorithm V^* there exists a simulator M such that

- Zero-knowledge: For every $x \in \mathcal{L}$, the transcript of $\langle P, V^* \rangle(x)$ (the output of the interactive machine V^* after interacting with P on common input x) and $M(x)$ (the output of M on input x) have the identical distribution.

3 Model of DCS

3.1 Syntax

The designated confirmer signature scheme consists of three parties, the signer S, the verifier V and the designated confirmer C. Since we disallow the confirmer C to convert the DCS signature produced by the signer S, the components of our DCS scheme have a little difference with the original ones.

Key Generation for S and C [$\mathtt{KGen_s}(1^k)$ and $\mathtt{KGen_c}(1^k)$]: Under the security parameter k, there is a probabilistic polynomial time (PPT) algorithm $\mathtt{KGen_s}(1^k)$ outputs a keypair of the signer S: (sk_s, vk_s). sk_s is S's signing key and vk_s is S's verification key. There is a PPT algorithm $\mathtt{KGen_c}(1^k)$ outputs a keypair of the confirmer C: (sk_c, pk_c). sk_c is C's private key and pk_c is C's public key.

DCS Signing [$\mathtt{DCSig}(m, pk_c; sk_s)$]: Given a message m, C's public key pk_c, the signer S produces a DCS signature by using his signing key sk_s: $\sigma \leftarrow \mathtt{DCSig}(m, pk_c; sk_s)$.

Confirmation Protocols [$\mathtt{Conf_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Conf_{c,v}}(m, \sigma, vk_s)$]: Both the signer S and confirmer C can run the confirmation algorithm with the verifier V to convince him that the DCS signature σ is valid with respect to the message m under the signer S's verification key vk_s. After performing these protocols, V outputs Accept or $\perp$. The confirmation protocols $\mathtt{Conf_{s,v}}$ and $\mathtt{Conf_{c,v}}$ should be complete and sound.

- Completeness. For all honest S, C and V, if the DCS $\sigma \leftarrow \mathtt{DCSig}(m, pk_c; sk_s)$, then $\mathtt{Accept} \leftarrow \mathtt{Conf_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Accept} \leftarrow \mathtt{Conf_{c,v}}(m, \sigma, vk_s)$ hold with overwhelming probability.
- Soundness. For all malicious S′, C′ and honest V, if $\sigma \leftarrow \mathtt{DCSig}(m, pk_j; sk_i)$, then $\mathtt{Accept} \leftarrow \mathtt{Conf_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Accept} \leftarrow \mathtt{Conf_{c,v}}(m, \sigma, vk_s)$ hold with negligible probability, where $vk_i \neq vk_s$ and $pk_j \neq pk_c$.

Disavowal Protocols: [$\mathtt{Disa_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Disa_{c,v}}(m, \sigma, vk_s)$]: Both the signer S and confirmer C can run the disavowal algorithm with the verifier V to convince him that the DCS signature σ is invalid with respect to the message m under the signer S's verification key vk_s. After performing these protocols, V outputs Accept or $\perp$. The disavowal protocols $\mathtt{Disa_{s,v}}$ and $\mathtt{Disa_{c,v}}$ should be complete and sound.

- Completeness. For all honest S, C and V, if the DCS $\sigma \leftarrow \mathtt{DCSig}(m, pk_j; sk_i)$, then $\mathtt{Accept} \leftarrow \mathtt{Disa_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Accept} \leftarrow \mathtt{Disa_{c,v}}(m, \sigma, vk_s)$ hold with overwhelming probability, where $vk_i \neq vk_s$ and $pk_j \neq pk_c$.
- Soundness. For all malicious S′, C′ and honest V, if $\sigma \leftarrow \mathtt{DCSig}(m, pk_c; sk_s)$, then $\mathtt{Accept} \leftarrow \mathtt{Disa_{s,v}}(m, \sigma, vk_s, pk_c)$ and $\mathtt{Accept} \leftarrow \mathtt{Disa_{c,v}}(m, \sigma, vk_s)$ hold with negligible probability.

3.2 Security Model

We follow the definitions in [10] and [15] (which both support the unified verification) to describe the syntax of our scheme above. However, in this proposal, we require that the confirmer $\mathtt{C}$ cannot convert a DCS σ into a publicly verifiable signature σ', such that any verifier $\mathtt{V}$ can verify the validity of σ' without the help of $\mathtt{S}$ or $\mathtt{C}$. Therefore, both the syntax and the security model are updated. In the syntax description, we delete the Extract algorithm from [10] and [15]. And in the following security model, we should add the formal model *Inconvertibility* to describe the security for the signer that although $\mathtt{C}$ can prove the validity of the DCS signature σ, he cannot convert σ into a standard signature σ'.

Let O_{DCSig} be the DCS signature signing oracle when the attacker $\mathcal{A}$ queries a DCS signature w.r.t. message m under a verification vk_s and a public key pk_c. The oracle O_{DCSig} returns a DCS signature $\sigma \leftarrow \mathtt{DCSig}(m, pk_c; sk_s)$. Let O_{Conf} be a confirmation oracle when $\mathcal{A}$ queries the validity of σ under vk_s. This oracle takes sk_s or sk_c as its auxiliary input to interact with the verifier to convince him that σ is valid w.r.t. m under vk_s. Let O_{Disa} be a disavowal oracle when $\mathcal{A}$ queries the invalidity of a fake DCS signature σ under vk_s. This oracle takes sk_s or sk_c as its auxiliary input to interact with the verifier to convince him that σ is invalid w.r.t. m under vk_s.

Definition 3 (Security for the signer-Unforgeability). *Let $\mathcal{F}$ be a PPT forger. Upon input vk_s, sk_c, pk_c, $\mathcal{F}$ can access to O_{DCSig}, $O^{\mathtt{S},\mathcal{F}}_{Conf}$ and $O^{\mathtt{S},\mathcal{F}}_{Disa}$ oracles adaptively. Finally, $\mathcal{F}$ outputs a DCS signature pair (m^*, σ^*) and plays the roles of* $\mathtt{S}$ *and* $\mathtt{C}$ *to execute the confirmation protocol with the challenger. $\mathcal{F}$* succeeds *if m^* is not queried in O_{DCSig} and the challenger outputs* Accept *after performing the confirmation protocol with $\mathcal{F}$. Let $\mathtt{Succ}^{\mathtt{unfor}}(\mathcal{F})$ denote the success of $\mathcal{F}$. We say a DCS scheme is secure against existential forgeability if $\Pr[\mathtt{Succ}^{\mathtt{unfor}}(\mathcal{F})]$ is negligible.*

Remark 1. *Unforgeability requires that no adaptive PPT adversary can forge a valid DCS signature on a fresh message on behalf of a signer although it corrupts the confirmer* $\mathtt{C}$. *Similar to [15], the oracles $O^{\mathtt{V},\mathcal{F}}_{Conf}$ and $O^{\mathtt{V},\mathcal{F}}_{Disa}$ are not necessary to be accessed by $\mathcal{F}$ in the above unforgeability game since the confirmer can be corrupted by $\mathcal{F}$. However, the condition that $\mathcal{F}$ wins the unforgeability game is slightly different with [10] and [15]. Since in both [10] and [15], they have* Extract *and* Verify *algorithms. The validity of the forgery can be checked by running* Extract *and* Verify *algorithms. However, in our scheme, we remove* Extract *and* Verify *algorithms. The validity of $\mathcal{F}$'s forgery can be convinced if $\mathcal{F}$ plays as a prover's (signer or confirmer) role to conduct the confirmation protocol.*

Definition 4 (Security for the signer-Inconvertibility). *Let $\mathcal{A}$ be a PPT attacker. Upon input vk_s, sk_c, pk_c, $\mathcal{A}$ can access to O_{DCSig}, $O^{\mathtt{S},\mathcal{A}}_{Conf}$ and $O^{\mathtt{S},\mathcal{A}}_{Disa}$ oracles adaptively. Finally, $\mathcal{A}$ outputs an ordinary signature pair (m, σ'). $\mathcal{A}$* succeeds *if $\mathtt{Veri}(m, \sigma', vk_s) = 1$, where* Veri *is an algorithm that anyone can input the verification key vk_s to check the validity of σ'. Let $\mathtt{Succ}^{\mathtt{Incon}}(\mathcal{A})$ denote the success of $\mathcal{A}$. We say a DCS scheme is secure against convertibility if $\Pr[\mathtt{Succ}^{\mathtt{Incon}}(\mathcal{A})]$ is negligible.*

Remark 2. *Inconvertibility requires that no adaptive PPT adversary can convert any DCS signature σ into an ordinary signature σ' such that σ' can be verified by anyone. We note this property is necessary for the signer's privacy. In the original DCS schemes, the confirmer has the capability to extract a DCS signature. However, it is not secure for the signer if the confirmer converts the DCS signature into an ordinary signature without signer's consent. Therefore, in our scheme, we disallow the confirmer to convert the DCS signature into a publicly verifiable signature.*

Definition 5 (Security for the confirmer-Invisibility). *Let $\mathcal{D}$ be a PPT distinguisher. Upon input vk_s and pk_c, $\mathcal{D}$ can access to O_{DCSig}, $O^{\mathtt{S},\mathcal{D}}_{Conf}$, $O^{\mathtt{C},\mathcal{D}}_{Conf}$, $O^{\mathtt{S},\mathcal{D}}_{Disa}$ and $O^{\mathtt{C},\mathcal{D}}_{Disa}$ oracles adaptively. Then, $\mathcal{D}$ outputs a fresh message m^*. The challenger tosses a coin $b \in \{0,1\}$ and when $b = 0$, $\mathcal{D}$ is given a DCS signature $\sigma^* \leftarrow \mathtt{DCSig}(m^*, pk_c; sk_s)$; while $b = 1$, $\mathcal{D}$ is given a random value from the signature space. After given the challenge DCS signature, $\mathcal{D}$ can continue to access the above oracles except the oracles for σ^*. Finally, $\mathcal{D}$ outputs its guess bit b'. $\mathcal{D}$* succeeds *if $b' = b$. Let $\mathtt{Succ}^{\mathtt{Invis}}(\mathcal{D})$ denote the success of $\mathcal{D}$. We say a DCS scheme is secure against visibility if $|\Pr[\mathtt{Succ}^{\mathtt{Invis}}(\mathcal{D})] - 1/2|$ is negligible.*

Remark 3. *The invisibility model actually requires that any verifier cannot check the validity of a DCS signature without the help from the signer or confirmer. The adversary is disallowed to corrupt* $\mathtt{S}$ *and* $\mathtt{C}$ *and also cannot query O_{Conf} and O_{Disa} oracles for the challenge signature σ^*. Otherwise, it is trivial for him to get the validity/invalidity of the DCS signature σ^*.*

Definition 6. *A designated confirmer signature scheme is* secure *if unforgeability, inconvertibility and invisibility hold.*

4 Construction

Setup. Choose two safe primes p, q and compute $n = pq$. Choose two multiplicative cyclic groups G, G_T of orders n that are associated to a bilinear pairing $\hat{e} : G \times G \rightarrow G_T$. G_q is the subgroup of G with order q. g and h are the random generators of G. $\hat{h}$ is the generator of G_q. $H : \{0,1\}^* \rightarrow \mathbb{Z}_n$ is a collision-resistant hash function. The system public parameter $para = (G, G_T, \hat{e}, n, g, h, \hat{h}, H)$.

Key Generation $\mathtt{KGen_s}(1^k)$ and $\mathtt{KGen_c}(1^k)$.

- For the signer $\mathtt{S}$: Choose random values $x, y \leftarrow \mathbb{Z}_n$ as his signing key sk_s and set the corresponding verification key $vk_s = g^x h^y$.

- For the confirmer C: His private key is $sk_c = q$ and the public key is $pk_c = \hat{h}$.

DCS Signing $\texttt{DCSig}(m, pk_c; sk_s)$. Suppose m is the message to be signed. Upon input m, sk_s and pk_c, the signer S chooses $r \leftarrow \{0,1\}^k$ and computes $\tau = H(m,r)$. Then S returns the designated confirmer signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, r)$, where $\sigma_1 = g^{\frac{1}{x+\tau}}$, $\sigma_2 = g^{\frac{y}{x+\tau}}\hat{h}^x$, $\sigma_3 = \hat{h}^{\frac{1}{x+\tau}}$.

Confirmation Protocols. Both the signer S and the confirmer C can confirm the validity of a signature σ with respect to (m,r). The confirmation algorithms for S and C are as follows.

- The confirmation $\texttt{Conf}_{\texttt{s,v}}(m, \sigma, vk_s, pk_c)$ for the signer S: S executes a zero-knowledge proof with any verifier V by using the witness (x,y) for the $\mathcal{NP}$ language $\mathcal{L}_s$:

$$\mathcal{L}_s = \Big\{(vk_s, pk_c, \tau, \sigma_1, \sigma_2) \mid \hat{e}(\hat{h}, h)^x = \frac{\hat{e}(g,g)\hat{e}(\sigma_2,h)}{\hat{e}(\sigma_1, vk_s g^\tau)} \wedge vk_s = g^x h^y\Big\}$$

  The common input of S and V is $(vk_s, pk_c, \tau, \sigma)$, where $\tau = H(m,r)$. The auxiliary input of the prover S is his witness (x,y). After this interaction, V is convinced that σ is valid on the message m under the signer S's verification key vk_s.

- The confirmation $\texttt{Conf}_{\texttt{c,v}}(m, \sigma, vk_s)$ for the confirmer C: When C performs the confirmation protocol with the verifier V, he first checks the consistency of σ_1 and σ_3. Then he executes a zero-knowledge proof with V.

  1) Check $\hat{e}(\sigma_1, \hat{h}) \overset{?}{=} \hat{e}(g, \sigma_3)$. If no, C declares `Failure` and quits; Otherwise, C turns to the step 2);
  2) Execute a zero-knowledge proof with V by using the witness q for the $\mathcal{NP}$ language $\mathcal{L}_c$:

$$\mathcal{L}_c = \Big\{(vk_s, \tau, \sigma_1, \sigma_2) \mid \hat{e}(\sigma_1, vk_s g^\tau)^q = \hat{e}(g,g)^q \hat{e}(\sigma_2, h)^q\Big\}$$

  The common input of C and V is (vk_s, τ, σ), where $\tau = H(m,r)$. The auxiliary input of the prover C is his witness q. After this interaction, V is convinced that σ is valid on the message m under the signer S's verification key vk_s.

Disavowal. Both the signer S and the confirmer C can disavow an invalid DCS signature σ w.r.t. (m,r). The disavowal algorithms for S and C are as follows.

- The disavowal $\texttt{Disa}_{\texttt{s,v}}(m, \sigma, vk_s, pk_c)$ for the signer S: S executes a zero-knowledge proof with any verifier V by using the witness (x,y) for the $\mathcal{NP}$ language $\mathcal{L}'_s$:

$$\mathcal{L}'_s = \Big\{(vk_s, pk_c, \tau, \sigma_1, \sigma_2) \mid \hat{e}(\hat{h}, h)^x \neq \frac{\hat{e}(g,g)\hat{e}(\sigma_2,h)}{\hat{e}(\sigma_1, vk_s g^\tau)} \wedge vk_s = g^x h^y\Big\}$$

  The common input of S and V is $(vk_s, pk_c, \tau, \sigma)$, the auxiliary input of the prover S is his witness (x,y). After this interaction, V is convinced that σ is invalid on the message m under the signer S's verification key vk_s.

- The disavowal $\texttt{Disa}_{\texttt{c,v}}(m, \sigma, vk_s)$ for the confirmer C: When C performs the disavowal protocol with the verifier V, he first checks the consistency of σ_1 and σ_3. Then he executes a zero-knowledge proof with V.

  1) Check $\hat{e}(\sigma_1, \hat{h}) \overset{?}{=} \hat{e}(g, \sigma_3)$. If no, C declares `Failure` and quits; Otherwise, C turns to the step 2);
  2) Execute a zero-knowledge proof by using the witness q for the $\mathcal{NP}$ language $\mathcal{L}'_c$:

$$\mathcal{L}'_c = \Big\{(vk_s, \tau, \sigma_1, \sigma_2) \mid \hat{e}(\sigma_1, vk_s g^\tau)^q \neq \hat{e}(g,g)^q \hat{e}(\sigma_2, h)^q\Big\}$$

  The common input of C and V is (vk_s, τ, σ), the auxiliary input of the prover C is his witness q. After this interaction, V is convinced that σ is invalid on the message m under the signer S's verification key vk_s.

Remark 4. *Our construction is based on the BB signature scheme and the BGN commitment scheme [2]. The public key of BB signature scheme is the partial public key of our scheme. In other words, BB signature σ_1 is a partial signature of DCS and the validity of DCS cannot be checked directly from the partial signature σ_1. We adopt the BGN commitment scheme to generate σ_2. When $\hat{h}$ is from G_q, the confirmer can use his private key q to convince the verifier that DCS is valid or not. Therefore, the confirmation/disavowal protocol is achieved. When $\hat{h}$ is from G, $g^{\frac{y}{x+\tau}}$ is perfectly hidden. Therefore, the invisibility is satisfied. σ_3 in our scheme is used to convince the verifier that an attacker cannot make another σ'_1 such that $\sigma'_1 \neq \sigma_1$ but $(\sigma'_1)^q = (\sigma_1)^q$. Otherwise, it fails to meet the invisibility. Assuming a challenge DCS σ^* is given. An attacker $\mathcal{D}$ can check the validity of σ^* by querying confirmer the validity of $\sigma'_1 = \sigma_1^* \hat{h}^s$ by selecting a random value s. This query is allowed in the security model of invisibility. In case of such attack, σ_3 is necessary. When $\mathcal{D}$ makes a confirmation/disavowal query, the confirmer first checks $\hat{e}(\sigma_1, \hat{h}) \overset{?}{=} \hat{e}(g, \sigma_3)$. If yes, the confirmer is convinced that $\mathcal{D}$ cannot make another σ'_1 such that $\sigma'_1 \neq \sigma_1$ but $(\sigma'_1)^q = (\sigma_1)^q$.*

Remark 5. *Our scheme is unextractable. The confirmer cannot convert the DCS into an original signature although he has the private key q. This property is good for the signer. Suppose in the original DCS scheme, the confirmer can extract a standard signature from a DCS signature. If the confirmer is malicious, he would make use of this capability to threaten the signer that he will* *convert the DCS signature into a publicly verifiable signature. While in our construction, given the DCS signature, the confirmer can only use his private key to convince the verifier that DCS is valid/invalid through zero-knowledge proof. He cannot convert it. Intuitively, the confirmer computes σ_2^q and only to obtain the $g^{\frac{y}{x+\tau}}$'s projection on G_p. We denote it by $\sigma_{2,p}$. Therefore, with the private key q, the confirmer only convinces the verifier that $\hat{e}(\sigma_{1,p}, vk_{s,p} \cdot g_p^\tau) = \hat{e}(g_p, g_p)\hat{e}(\sigma_{2,p}, h_p)$. That is $\sigma_{1,p}$, $\sigma_{2,p}$ and $vk_{s,p}$ satisfy the verification equation where $vk_{s,p}$ is vk_s's projection on G_p. It is not publicly verifiable signature under the signer* $\mathtt{S}$*'s verification key vk_s.*

Remark 6. *Zero-knowledge protocol is the necessary tool in building a DCS scheme. In DCS scheme, we require that the signer should control the verifiability of a signature. That means the signer or confirmer convinces the verifier the validity of a signature but the verifier cannot convince the others. Therefore, zero-knowledge proof system should be applied into the DCS scheme naturally. In our confirmation/disavowal protocol, the zero-knowledge proofs can be implemented by running Σ-protocols with special soundness and perfect special honest-verifier zero-knowledge (HVZK) for the equality/inequality of discrete logarithms. It can be constructed by using the techniques in [5] and [11].*

5 Performance and Property Comparison

To the best of our knowledge, there are two constructions [10, 15] in the literature support the unified verification. That is the signer also has the capability to run the disavowal protocol to deny invalid signatures. Therefore, we compare our scheme with [10, 15].

For the security, Wang's scheme [15] relies on the random oracles while Huang's scheme [10] and our scheme are secure without random oracles.

We then analyze the communication and computation costs among the three constructions. Since inherited from the shortness of "multi-generator" instantiation of programmable hash function, the verification key of the signer in Huang's scheme involves 162 elements of the bilinear group $\mathbb{G}$ (whose order is prime p and it is about 20 bytes for 80-bit security). In Wang's scheme, the signer's verification key requires 1 element of the bilinear group $\mathbb{G}$ and the verification key of signer in our scheme requires 1 element of the bilinear group G (whose order is RSA composite n and it is about 1024 bits for 80-bit security). For the size of the DCS signature, the DCS signature in Huang's DCS scheme contains 3 elements of $\mathbb{G}$, in Wang's scheme needs 2 elements of $\mathbb{G}$ and 2 elements of $\mathbb{Z}_p$ and in our scheme needs 3 elements of G and a randomness r. During the generation of DCS signature, both [10] and [15] take 3 exponentiations and our scheme takes 4 exponentiations.

Finally, let us discuss the properties of the three schemes. All the three schemes support the unified verification. Since the DCS signatures in [10] and [15] are produced using the encryption format, the confirmer can extract the original signature by decrypting the DCS signature using his private key. Therefore, we say [10] and [15] cannot meet the property of inconvertibility. While our scheme uses the BGN commitment to generate the DCS signature, the confirmer has the capability to confirm/disavow valid/invalid signatures but cannot extract them. Therefore, our scheme achieves inconvertibility.

Detailed comparison is shown is Table 1.

Remark 7. *Since we adopt the bilinear group for composite order n, it seems our DCS signature takes much larger size than [10] and [15]. It is also the weakness of constructions built on the composite order group.*

6 Security

Unforgeability. Unforgeability essentially states that no PPT forger $\mathcal{F}$ can generate a DCS signature on behalf of the signer $\mathtt{S}$ on a fresh message m^* even though it can access to O_{DCSig}, $O_{Conf}^{\mathtt{S},\mathcal{F}}$ and $O_{Disa}^{\mathtt{S},\mathcal{F}}$ oracles adaptively. Additionally, $\mathcal{F}$ owns the private-public keypair (sk_c, pk_c) of the confirmer $\mathtt{C}$ and $\mathcal{F}$ cannot query O_{DCSig} on m^*. Our unforgeability is reduced to the security of BB signature which is based on the Strong Diffie-Hellman assumption without the random oracles. Formally,

Theorem 2. *Our scheme is unforgeable if BB signature scheme is secure against existential forgery and the zero-knowledge protocol for the language $\mathcal{L}_s$ satisfies soundness and zero-knowledge.*

Proof. In the unforgeability game, $\mathcal{F}$ is the forger who violates the unforgeability of our DCS scheme. $\mathcal{C}$ is $\mathcal{F}$'s challenger whose goal is to forge a BB signature under the composite order group. $\mathcal{C}$ is given the group order n and its factorization $n = pq$; the description of the group G and the BB signature public key (g, g^x), where g is the generator of G. $\mathcal{C}$ randomly chooses $s_1 \leftarrow \mathbb{Z}_n$, $s_2 \equiv 0 \bmod p$ and $y \leftarrow \mathbb{Z}_n$. $\mathcal{C}$ sets $h = g^{s_1}$, $\hat{h} = g^{s_2}$ and $vk_s = g^x h^y$. $\mathcal{C}$ gives $g, h, vk_s = g^x h^y, sk_c = q, pk_c = \hat{h}$ to $\mathcal{F}$. $\mathcal{C}$'s simulation for $\mathcal{F}$ is as follows.

$O_{DCSig}(m, vk_s, pk_c)$. When $\mathcal{F}$ queries DCS signing oracles on (m, vk_s, pk_c), $\mathcal{C}$ queries its BB signature challenger to return a BB signature on message (m, r): $\sigma_{BB} = g^{\frac{1}{x+\tau}}$, where $\tau = H(m, r)$. Then $\mathcal{C}$ produces $\sigma_2 = \sigma_{BB}^y \hat{h}^x = \sigma_{BB}^y (g^x)^{s_2}$, $\sigma_3 = \sigma_1^{s_2}$. The DCS signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, r)$ is as the reply of $O_{DCSig}(m, vk_s, pk_c)$, where $\sigma_1 = \sigma_{BB}$.

$O_{Conf}^{\mathtt{S},\mathcal{F}}(m, r, \sigma, vk_s, pk_c)$. When $\mathcal{F}$ makes such queries, $\mathcal{C}$ checks the validity of DCS signature σ by verifying $\hat{e}(\hat{h}, g^x)^{s_1} \overset{?}{=} \frac{\hat{e}(g,g)\hat{e}(\sigma_2,h)}{\hat{e}(\sigma_1, vk_s \cdot g^\tau)}$, where $\tau = H(m, r)$. If yes, $\mathcal{C}$ uses the trapdoor s_1 to simulate the language $\mathcal{L}_s$; Otherwise, $\mathcal{C}$ turns to run $O_{Disa}^{\mathtt{S},\mathcal{F}}(m, r, \sigma, vk_s, pk_c)$. Since

Table 1: Comparison with [10] and [15]

Scheme	Size					Computation	ROM	InCon	UV
	vk_s	sk_s	pk_s	sk_c	DCS	DCS			
[10]	$162\mathbb{G}$	$1\mathbb{Z}_p$	$1\mathbb{G}$	$1\mathbb{Z}_p$	$3\mathbb{G}$	$3\mathbf{e}$	No	No	Yes
[15]	$1\mathbb{G}$	$1\mathbb{Z}_p$	$1\mathbb{G}$	$1\mathbb{Z}_p$	$2\mathbb{G}+2\mathbb{Z}_p$	$3\mathbf{e}$	Yes	No	Yes
Ours	$1G$	$2\mathbb{Z}_n$	$1G$	$1\|q\|$	$3G+k$	$4\mathbf{e}$	No	Yes	Yes

$\mathbf{e}$: Exponentiation; InCon: Inconvertiability; UV: Unified Verification;
$|q| \doteq 510$bits; k: Security Parameter.

zero-knowledge protocol for the language $\mathcal{L}_s$ satisfies the zero-knowledge, there exists a simulator (by using the trapdoor) can simulate the transcript communicated between the signer and verifier without the witness and the simulated transcript is indistinguishable from the real proof. Therefore, the view of $\mathcal{F}$ in $\mathcal{C}$'s simulation for $O^{\mathtt{S},\mathcal{F}}_{Conf}(m,r,\sigma,vk_s,pk_c)$ equals to the real ones. Otherwise, it violates the zero-knowledge of ZK proof system. Therefore, $\mathcal{C}$'s simulation for $O^{\mathtt{S},\mathcal{F}}_{Conf}(m,r,\sigma,vk_s,pk_c)$ is perfect although it has no witness x.

$O^{\mathtt{S},\mathcal{F}}_{Disa}(m,r,\sigma,vk_s,pk_c)$. When $\mathcal{F}$ makes such queries, $\mathcal{C}$ uses the trapdoor s_1 to simulate the language $\mathcal{L}'_s$ as the description of performing $O^{\mathtt{S},\mathcal{F}}_{Conf}(m,r,\sigma,vk_s,pk_c)$.

Finally, $\mathcal{F}$ challenges fresh (m^*,r^*) and outputs a DCS signature $\sigma^* = (\sigma_1^*,\sigma_2^*,\sigma_3^*,r^*)$ w.r.t. (m^*,r^*). Assuming $\mathcal{F}$ wins the unforgeability game, which means that $\mathcal{F}$ plays the roles of the signer and confirmer to perform the confirmation protocol to convince the challenger $\mathcal{C}$ that the language $\mathcal{L}_s$ which states $\hat{e}(\hat{h},h)^x = \frac{\hat{e}(g,g)\hat{e}(\sigma_2,h)}{\hat{e}(\sigma_1,vk_s g^{\tau})}$. Due to the soundness of the confirmation protocol, the equation $\hat{e}(\sigma_1^*,vk_s g^{\tau^*})\hat{e}(\hat{h},h)^x = \hat{e}(g,g)\hat{e}(\sigma_2^*,h)$ must hold, where $\tau^* = H(m^*,r^*)$. $\mathcal{C}$ furthermore checks if $\hat{e}(\sigma_1^*,h^y)\hat{e}(g^x,h)^{s_2} = \hat{e}(\sigma_2^*,h)$ holds, then $\mathcal{C}$ obtains $\hat{e}(\sigma_1^*,g^x g^{\tau^*}) = \hat{e}(g,g)$ which means $\mathcal{C}$ produces a valid BB signature forgery σ_1^* w.r.t. fresh (m^*,r^*) under the verification key $vk_s = g^x$. □

Invisibility. Generally speaking, invisibility states that no PPT distinguisher $\mathcal{D}$ can check the validity of a DCS signature without the help of the signer $\mathtt{S}$ or the confirmer $\mathtt{C}$. During the invisibility game, $\mathcal{D}$ can access to O_{DCSig}, $O^{\mathtt{S},\mathcal{D}}_{Conf}$, $O^{\mathtt{C},\mathcal{D}}_{Conf}$, $O^{\mathtt{S},\mathcal{D}}_{Disa}$ and $O^{\mathtt{C},\mathcal{D}}_{Disa}$ oracles adaptively. However, after given the challenge DCS signature σ^*, $\mathcal{D}$ cannot query $O_{Conf}(m,r,\sigma^*,vk_s,pk_c)$ and $O_{Disa}(m,r,\sigma^*,vk_s,pk_c)$. Of course, $\mathcal{F}$ cannot corrupt both signer and confirmer. Otherwise, it is trivial for $\mathcal{D}$ to check the validity of σ^* by using the private key of the signer or confirmer. Our invisibility relies on the Subgroup Decision assumption (see Definition 2). Formally,

Theorem 3. *Our scheme is invisible if Subgroup Decision assumption holds and the zero-knowledge proof system for the language $\mathcal{L}_c$ satisfies zero-knowledge.*

Proof. Assuming $\mathcal{D}$ is a PPT distinguisher against the invisibility of our DCS scheme. $\mathcal{C}$ is $\mathcal{D}$'s challenger whose goal is to solve the subgroup decision assumption. $\mathcal{C}$ is given a group element $\hat{h}$ and tries to decide if $\hat{h}$ is from the group G whose order is composite $n = pq$ or from the subgroup G_q of G.

After obtaining the description of the group G, $\mathcal{C}$ randomly chooses generators g,h of G. $\mathcal{C}$ gives the public parameter *para* of the DCS scheme to $\mathcal{D}$: $para = (G,G_T,\hat{e},n,g,h,\hat{h},H)$.

For the key generation algorithm $\mathtt{KGen_s}(1^k)$, $\mathcal{C}$ randomly chooses x,y as the signer's signing key sk_s, and $vk_s = g^x h^y$; For the key generation algorithm $\mathtt{KGen_c}(1^k)$, $\mathcal{C}$ sets $\hat{h}$ as the confirmer's public key pk_c. Finally, $\mathcal{C}$ gives (vk_s,pk_c) to $\mathcal{D}$.

It is trivial to simulate the DCS signing algorithm $\mathtt{DCSig}(m,pk_c;sk_s)$. $\mathcal{C}$ produces the DCS signature by using the signer's signing key (x,y).

It is also trivial for $\mathcal{C}$ to complete the signer's confirmation/disavowal simulation since $\mathcal{C}$ has the witness (x,y) to perform the zero-knowledge protocol for the language $\mathcal{L}_s/\mathcal{L}'_s$. However, for the confirmer's confirmation/disavowal simulation, $\mathcal{C}$ does not have the factorization q of n. Therefore, it cannot use the witness q to generate the zero-knowledge proof for $\mathcal{L}_c/\mathcal{L}'_c$. $\mathcal{C}$'s simulation for confirmation/disavowal is as follows. When (m,σ,vk_s,pk_c) is queried, $\mathcal{C}$ checks the validity of the tuple (m,σ,vk_s,pk_c) by using (x,y). If it is valid with respect to m under vk_s, $\mathcal{C}$ simulates the zero-knowledge proof for the language $\mathcal{L}_c$ without the witness q. Therefore, a simulated proof for $\mathcal{L}_c$ is performed between the prover $\mathcal{C}$ and the verifier $\mathcal{D}$. If it is invalid with respect to m under vk_s, $\mathcal{C}$ simulates the zero-knowledge proof for the language $\mathcal{L}'_c$ without the witness q. Therefore, a simulated proof for $\mathcal{L}'_c$ is performed between the prover $\mathcal{C}$ and the verifier $\mathcal{D}$. Due to the zero-knowledge of zero-knowledge protocol, the simulated proof for $\mathcal{L}_c$ or $\mathcal{L}'_c$ is indistinguishable by $\mathcal{D}$. In other words, $\mathcal{D}$ has the same view in the simulated confirmation/disavowal protocol and the real confirmation/disavowal protocol. Therefore, $\mathcal{C}$'s simulation of confirmation/disavowal protocol is perfect.

Finally, $\mathcal{D}$ chooses a fresh message m^*. $\mathcal{C}$ tosses a coin $b \in \{0,1\}$. If $b = 0$, $\mathcal{C}$ generates a challenge DCS σ^* following the $\texttt{DCSig}$ algorithm. If $b = 1$, $\mathcal{C}$ randomly chooses a signature σ_R^* from the signature space. $\mathcal{D}$ also can make O_{DCSig}, $O_{Conf}^{\texttt{S},\mathcal{D}}$, $O_{Conf}^{\texttt{C},\mathcal{D}}$, $O_{Disa}^{\texttt{S},\mathcal{D}}$ and $O_{Disa}^{\texttt{C},\mathcal{D}}$ queries except $O_{Conf}(m^*, \sigma^*, vk_s, pk_c)$ and $O_{Disa}(m^*, \sigma^*, vk_s, pk_c)$. Now, let us discuss the advantage that $\mathcal{D}$ wins in the invisibility game. If $\hat{h}$ is chosen uniformly from G_q, $\mathcal{D}$'s environment is exactly as specified in the real game. If $\hat{h}$ is chosen uniformly from G, the advantage that $\mathcal{D}$ wins invisibility game is negligible even if $\mathcal{D}$ is computationally unbounded since $g^{\frac{y}{x+\tau}}$ is perfectly hidden in σ_2. Therefore, if the advantage that $\mathcal{D}$ breaks invisibility is non-negligible, it shows that $\hat{h}$ is from G_q. Therefore, $\mathcal{D}$'s success implies subgroup decision assumption is solved. □

Inconvertibility. The confirmer $\texttt{C}$ cannot convert the DCS signature σ into an original signature σ'. Intuitively, for a DCS signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, r) = (g^{\frac{1}{x+\tau}}, g^{\frac{y}{x+\tau}}\hat{h}^x, \hat{h}^{\frac{1}{x+\tau}}, r)$, $\texttt{C}$ must obtain $g^{\frac{y}{x+\tau}}$ from σ_2 such that $(g^{\frac{1}{x+\tau}}, g^{\frac{y}{x+\tau}})$ is publicly verifiable under the verification key vk_s, i.e. by checking $\hat{e}(g^{\frac{1}{x+\tau}}, vk_s \cdot g^\tau) = \hat{e}(g,g)\hat{e}(g^{\frac{y}{x+\tau}}, h)$. However, $\texttt{C}$ cannot get $g^{\frac{y}{x+\tau}}$ from σ_2 even though he has the private key q. $\texttt{C}$ raises σ_2 to power q to remove $\hat{h}^x$ and only to obtain $g_p^{\frac{y}{x+\tau}}$, which equals to $g^{\frac{y}{x+\tau}}$'s projection on G_p. Therefore, with q, $\texttt{C}$ only convinces the verifier that $\hat{e}(\sigma_{1,p}, vk_{s,p} \cdot g_p^\tau) = \hat{e}(g_p, g_p)\hat{e}(\sigma_{2,p}, h_p)$, where $vk_{s,p}$ is vk_s's projection on G_p. It is not publicly verifiable signature under the signer's verification key vk_s.

7 Conclusion

In this paper, we construct a new designated confirmer signature without random oracles. This new construction achieves the full verification. That is, it enables the signer to deny any invalid signatures. Different from the traditional DCS schemes, our scheme disallows the confirmer (a semi-trusted third party) to convert a DCS signature into a publicly verifiable signature. This property is good for the signer's privacy. Our scheme combines BB signature and BGN commitment to achieve it. However, our DCS signature takes a large size due to the composite order of the group. It is significant to realize inconvertibility without bilinear group of composite order.

Acknowledgments

This work is supported by Fundamental Research Funds for the Central Universities (No.ZYGX2011J068) and the NSFS (No.61133016 and No.61003230).

References

[1] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Eurocrypt '04*, pp. 56–73, Interlaken, Switzerland, 2004.

[2] D. Boneh, E. J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of Cryptography Conference*, pp. 325–341, Cambridge, MA, USA, 2005.

[3] N. Chandran, J. Groth, and A. Sahai, "Ring signatures of sub-linear size without random oracles," in *International Colloquium on Automata, Languages and Programming (ICALP '07)*, pp. 423–434, Wroclaw, Poland, 2007.

[4] D. Chaum, "Designated confirmer signatures," in *Eurocrypt '94*, pp. 86–91, Perugia, Italy, 1995.

[5] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *Crypto '92*, pp. 89–105, Santa Barbara, USA, 1993.

[6] D. Chaum and H. van Antwerpen, "Undeniable signatures," in *Crypto '89*, pp. 212–216, Santa Barbara, USA, 1990.

[7] S. D. Galbraith and W. Mao, "Invisibility and anonymity of undeniable and confirmer signatures," in *Cryptographers' Track at the RSA (CT-RSA '03)*, pp. 80–97, San Francisco, USA, 2003.

[8] C. Gentry, D. Molnar, and Z. Ramzan, "Efficient designated confirmer signatures without random oralces or general zero-knowledge proofs," in *Asiacrypt '05*, pp. 662–681, Chennai, India, 2005.

[9] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proofs," in *ACM Symposium on Theory of Computing (STOC '85)*, pp. 291–304, Providence, Rhode Island, USA, 1985.

[10] Q. Huang, D. S. Wong, and W. Susilo, "A new construction of designated confirmer signature and its application to optimistic fair exchange," in *Pairing-based Cryptography (Pairing '10)*, pp. 41–61, Ishikawa, Japan, 2010.

[11] K. Kurosawa and S. H. Heng, "3-move undeniable signature scheme," in *Eurocrypt '05*, pp. 181–197, Aarhus, Denmark, 2005.

[12] M. Michels and M. A. Stadler, "Generic constructions for secure and efficient confirmer signature scheme," in *Eurocrypt '98*, pp. 406–421, Espoo, Finland, 1998.

[13] T. Okamoto, "Designated confirmer signatures and public-key encryption are equivalent," in *Crypto '94*, pp. 61–74, Santa Barbara, USA, 1994.

[14] G. Wang, J. Baek, D. S. Wong, and F. Bao, "On the generic and efficient constructions of secure designated confirmer signatures," in *Public Key Cryptography (PKC '07)*, pp. 43–60, Beijing, China, 2007.

[15] G. Wang, F. Xia, and Y. Zhao, "Designated confirmer signatures with unified verification," in *IMA International Conference on Cryptography and Coding (IMACC '11)*, pp. 469–495, Oxford, UK, 2011.

[16] B. Wei, F. Zhang, and X. Chen, "A new type of designated confirmer signatures for a group of individuals," *International Journal of Network Security*, vol. 7, no. 2, pp. 293–300, 2008.

Shengke Zeng is an Assistant Professor at the School of Mathematics and Computer Engineering, Xihua

University. She received her Ph.D degree from University of Electronic Science and Technology of China (UESTC) in 2013. Her research interests include: Cryptography and Network Security.

Hu Xiong is an Assistant Professor at the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC). He received his Ph.D degree from University of Electronic Science and Technology of China, 2009. His research interests include: Cryptography and Network Security.