Identity-Based Directed Signature Scheme from Bilinear Pairings

Xun Sun[1], Jian-hua Li[1,2], Gong-liang Chen[2], and Shu-tang Yang[1,2]

[1] Department of Electronic Engineering, Shanghai Jiao Tong University
[2] School of Information Security Engineering, Shanghai Jiao Tong University
xun.sun.cn@gmail.com

Abstract. In a directed signature scheme, a verifier can exclusively verify the signatures designated to himself, and shares with the signer the ability to prove correctness of the signature to a third party when necessary. Directed signature schemes are suitable for applications such as bill of tax and bill of health. This paper studies directed signatures in the identity-based setting. We first present the syntax and security notion that includes unforgeability and invisibility, then propose a concrete identity-based directed signature scheme from bilinear pairings. We then prove our scheme existentially unforgeable under the computational Diffie-Hellman assumption, and invisible under the decisional Bilinear Diffie-Hellman assumption, both in the random oracle model.

Key words: Bilinear pairings; Directed signature; Existential unforgeability; Identity-based cryptography; Invisibility

1 Introduction

In traditional digital signature schemes, anyone in possession of a signature σ can tell whether it has been generated by a particular signer on a message M. This property is usually regarded as the basis for the non-repudiation aspect of digital signatures. However this public-verifiable property is not desirable when the signed message contains a sensitive agreement between two companies or private information.

To solve this problem, Chaum and van Antwerpen [8] introduced the concept of undeniable signatures. In an undeniable signature scheme, one party can verify a signature only by interaction with the legitimate signer through a confirmation protocol. Therefore the signer can control when and by whom his signatures can be verified. An alternative approach to undeniable signatures is the designated confirmer signature introduce by Chaum in 1994 [9]. In such scheme, the signer designates the ability to acknowledge (confirm) a signature to a trusted third party called confirmer, to ensure that the signature can still be verified when the original signer gets offline. Other signature schemes with controlled verifiability include limited verifier signatures [2, 10][3] and designated verifier signatures [18].

[3] The limited verifier signature scheme of [2] has been broken by [31].

These schemes mainly focus on protecting the privacy of the signer. However consider the following situation:

A tax office Alice has issued Bob a bill of tax, in the form of Alice's digital signature. Bob then wants to exclusively verify this signature without others being able to check its validity, because otherwise his financial status is exposed. Here any other party can see the content of the signed message, but can not be sure whether the tax bill is valid or legally bind the tax bill to Alice or Bob. Bob needs to prove his ratal to other authorities when necessary. Alice also shares the ability and responsibility to acknowledge a bill of tax because Bob, as an individual, may not be always available to do this.

The aforementioned signature systems with controlled verifiability don't fully address this situation because they either lack the option of the receiver's direct verification (in undeniable, designated confirmer signatures), don't allow proving validity of signature to others (in designated verifier signatures), or allow only the receiver to acknowledge the signature to a third party (in limited verifier signatures).

One solution to this problem is the directed signatures suggested by Lim and Lee in [23]. In a directed signature scheme, any signature is generated for a designated verifier, who can directly verify the signature while others know nothing on its validity. In addition, at the time of trouble or if necessary, both the signer and the designated verifier can prove to a third party that the signature is valid. As previously suggested [23, 26], directed signature schemes are suitable whence signed messages are personally or commercially sensitive to and/or somewhat obligatory on the designated receiver.

As an alternative, one may also apply signcryption schemes. The notion of signcryption was first proposed by Zheng [32] to perform the functionality of signature and encryption simultaneously, and more efficiently than the signature-then-encryption approach. Signcryption schemes can achieve directed verifiability by encrypting the message and the signature under the designated verifier's public key, thus only the designated verifier can retrieve the message and subsequently verify the signature. Almost all signcryption schemes can be extended to support public verification by publishing the message and the signature embedded in the ciphertext. The primary difference between directed signatures and signcryption is that, directed signatures do not protect message confidentiality, while signcryption schemes do. We believe that the additional encryption operation can usually cause efficiency overhead. Indeed, we show in Section 5 that, both being derived from the modified SOK-IBS scheme [3], our proposed directed signature scheme is more efficient than the signcryption scheme of [21]. Therefore directed signatures are more suitable for the application we considered above.

The first directed signature scheme was proposed by Lim and Lee [23] based on the Guillou-Quisquater signature scheme [16]. In 2004, Lal and Kumar [20] suggested another scheme based on Schnorr's signature. However, no formal model was present in [23] and [20]. In 2005, Laguillaumie *et al.* [19] studied the universally convertible directed signatures and presented a concrete scheme which is provably secure in the random oracle model [4]. In 2006, Lu and Cao [25] independently presented a formal model for directed signatures, and proposed such a scheme based on the RSA assumption. In 2007, Lu *et al.* [26] studied the notion of threshold directed signatures, and presented a (t, n) threshold directed signature scheme from bilinear pairings, which they proved existentially unforgeable based on the computational Diffie-Hellman (CDH) assumption.

All the above directed signature schemes work in the Public Key Infrastructure (PKI) based setting, where the public key is usually a "random" string that is unrelated to the user's identity. To bind the public key to its legitimate owner, a certificate authority (CA) needs to digitally sign a certificate claiming this relationship between the public key and the user. As a result, any verifier must obtain the valid certificate before performing signature verification. Nowadays, certificate management (including revocation, storage and distribution) and the computational cost of certificate verification incur the main complaint against traditional public key cryptosystems. To eliminate the burden of certificate management, Shamir introduced the notion of identity-based cryptography [27]. In an identity-based cryptosystem, a user's public key is just his publicly available identity (e.g. real name, email address, or IP address), hence no extra effort is necessary for ensuring the authenticity of a public key, the complexity of the certificate management is released. However, the study of directed signatures in the identity-based setting is far from satisfactory. The only identity-based directed signature scheme that the authors are aware of was proposed by Wang [28] in 2005. However, there is neither a formal model nor rigorous security proof in [28]. Besides, it also does not support public verification, which is useful for applications such as bill of tax since one may be required to prove his ratal in real life occasionally.

To fill this gap, in this paper, we study directed signatures in the identity-based setting [27, 5]. In particular, we regard our contribution as follows:

– We define the syntax for identity-based directed signature (IB-DS) and propose its security notion, which includes both existential unforgeability and invisibility.
– We propose an identity-based directed signature scheme from bilinear pairings, and prove (in the random oracle model) the scheme existentially unforgeable and invisible based on the CDH assumption and the decisional Bilinear Diffie-Hellman (DBDH) assumption, respectively. Our scheme is an extension of the modified Sakai-Ogishi-Kasahara identity-based signature (SOK-IBS) scheme due to Bellare *et al.* [3]. The security analysis of our scheme develops based on the work of Libert and Quisquater [21], and deals with some subtleties incurred by our extension carefully.

We mention that there exists a generic construction of identity-based signature (IBS) schemes from any PKI-based standard signature $\mathcal{S} = (\texttt{S.KeyGen}, \texttt{S.Sign}, \texttt{S.Verify})$. The master public/secret key (mpk, msk) is a pair of matching public/secret keys of the scheme $\mathcal{S}$. To generate a user private key corresponding to identity ID, the private key generator (PKG) runs $\texttt{S.KeyGen}$ to obtain (pk, sk) and computes $cert$ by signing the message $ID||pk$ using msk. Here $cert$ serves as a "certificate" binding *pk* to ID. In the signing phase of the IBS scheme, the user uses *sk* to sign the message. The identity-based signature then consists of this signature along with *cert* and *pk*. In [3], Bellare *et al.* formalized this construction and showed that, the resulting IBS is secure if $\mathcal{S}$ is secure. In [14], this method is extended to IBS schemes with additional properties (e.g., blind signatures, proxy signatures, and threshold signatures). However, this method does not work properly for directed signatures, because the signer should interact with the designated verifier before signing, in order to obtain his certified public key. But in our model of IB-DS schemes, there should not be any interaction between the signer and the verifier in the signing phase because it suffices to know the identity of the designated verifier.

The rest of the paper is organized as follows. We first review the bilinear pairings and the complexity assumptions, on which our scheme is based, in Section 2. The syntax and security notion of identity-based directed signature schemes are given in Section 3. We then present our identity-based directed signature scheme in Section 4 with security proof. In Section 5, we compare our scheme with related schemes in terms of efficiency and security. Section 6 concludes this paper.

2 Preliminaries

We now briefly review the properties of bilinear pairings and the complexity assumptions on which our scheme is based.

2.1 Bilinear Pairings

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be additive and multiplicative groups of prime order q, respectively. Let P be a generator of $\mathbb{G}_1$. A bilinear pairing is an *efficiently computable* map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, with the following properties:

- *Bilinearity*: For all $a, b \in \mathbb{Z}_q$, $e(aP, bP) = e(P, P)^{ab}$.
- *Non-degeneracy*: $e(P, P) \neq 1_{\mathbb{G}_2}$.

Bilinear pairings have found huge application in design of cryptosystems. See [5, 29, 24, 30] for examples and a survey due to Dutta *et al.* [12] for detailed descriptions.

2.2 Computational Diffie-Hellman Assumption

Let $\mathbb{G}_1, q, P$ be as described in Section 2.1. The challenger selects $a, b \in \mathbb{Z}_q^*$ at random from $\mathbb{Z}_q^*$ and outputs (P, $A = aP$, $B = bP$). The computational Diffie-Hellman (CDH) problem is to find abP.

Definition 1 (CDH Assumption). *The (ϵ, t)-CDH assumption holds in group $\mathbb{G}_1$ if there is no algorithm which runs in time at most t and solves the CDH problem with probability at least ϵ.*

2.3 Decisional Bilinear Diffie-Hellman Assumption

For the same setting described in Section 2.1, the challenger selects $a, b, c, z \in \mathbb{Z}_q^*$ at random and flips a fair coin $T \in \{0, 1\}$. If $T = 1$ the challenger outputs the tuple (P, $A = aP$, $B = bP$, $C = cP$, $Z = e(P, P)^{abc}$); otherwise it outputs (P, $A = aP$, $B = bP$, $C = cP$, $Z = e(P, P)^z$). The decisional Bilinear Diffie-Hellman (DBDH) problem is to output T. The advantage of a DBDH solver $\mathcal{B}$ is defined as

$$\begin{aligned} Adv_{\mathcal{B}} =& |Pr[\mathcal{B}(P, aP, bP, cP, Z) = 1 | Z = e(P, P)^{abc}] \\ & - Pr[\mathcal{B}(P, aP, bP, cP, Z) = 1 | Z = e(P, P)^z]|. \end{aligned}$$

Definition 2 (DBDH Assumption). *The (ϵ, t)-DBDH assumption holds if there is no algorithm which runs in time at most t and solves the DBDH problem with non-negligible advantage at least ϵ.*

3 Identity-Based Directed Signature Schemes

In an identity-based directed signature (IB-DS) scheme, a private key generator (PKG) holds a master key and issues private keys for the users. The latter then uses the private key to generate digital signature which is designated to a particular verifier, using the verifier's identity. Formally, an IB-DS scheme is defined by the following probabilistic polynomial-time (PPT) algorithms:

- System initialization (`Setup`): The PKG generates the system parameters `params` and the master key x, `params` are made public, while x is kept secret. `params` are implicit input to all the following algorithms.
- Key extraction (`Extract`): Given an identity ID and the master key x, the PKG computes the private key d_{ID} and sends it to the corresponding user through a secret channel.
- Signature generation (`Sign`): On input signer's identity ID_S, the verifier's identity ID_V, a message M and the private key d_{ID_S}, the signer ID_S generates his signature σ on M designated to ID_V.
- Direct verification (`DVerify`): Given signer's identity ID_S, verifier's identity ID_V and the corresponding private key d_{ID_V}, a message M and a signature σ, this algorithm checks the validity of σ to output 1 (`valid`) or 0 (`invalid`).
- Public verification (`PVerify`): On input signer's identity ID_S, verifier's identity ID_V, a message M, and a purported signature σ, a third party T, with an Aid provided by ID_S or ID_V, outputs 1 if σ is valid, and 0 otherwise.

Remark 1. A relaxation of the *`PVerify`* procedure in our definition compared to that of [25] and [26] is that, the Aid is not required to be verifiable. We believe that this relaxation is not a problem in real-world applications, as long as it is computationally infeasible for an adversary to forge a 'correct' Aid, which is the case for all the directed signature schemes proposed so far.

3.1 Security Notions

The security of IB-DS schemes consists of two aspects: *unforgeability* and *invisibility.*

Unforgeability. The standard security notion for digital signature schemes is existential unforgeability against adaptively chosen message attack [15]. Cha and Cheon generalized this notion to identity-based signature schemes [7]. [19] and [25] independently formalized this notion for directed signatures under their respective settings. Developed in the same line with [15, 7, 25], existential unforgeability of IB-DS schemes is defined by the following game between a challenger and a PPT attacker $\mathcal{A}$:

Setup. The challenger runs the `Setup` algorithm of the IB-DS scheme to obtain the public parameters `params` and the master secret. It then gives `params` to $\mathcal{A}$ and keeps the master secret to itself.

Queries. $\mathcal{A}$ adaptively makes a number of different queries to the challenger. Each query can be one of the following.

- Extract query. $\mathcal{A}$ requests the private key of any identity ID. The challenger runs the `Extract` algorithm on ID and forwards the output d_{ID} to $\mathcal{A}$.
- Sign query. $\mathcal{A}$ requests the signature of a signer ID_S to a designated verifier ID_V on message M. The challenger first runs the `Extract` algorithm to obtain a private key d_{ID_S} of ID_S, then runs the `Sign` algorithm on ID_S, ID_V, M and d_{ID_S} to obtain a signature σ, which is forwarded to $\mathcal{A}$.
- DVerify query. $\mathcal{A}$ submits (ID_S, ID_V, M, σ) to the challenger. The challenger first extracts ID_V's private key d_{ID_V}, then uses d_{ID_V} to verify the signature. If the signature is valid, the challenger returns 1 (`valid`) to $\mathcal{A}$, otherwise it returns 0 (`invalid`).
- PVerify query. $\mathcal{A}$ submits (ID_S, ID_V, M, σ) to the challenger. The challenger returns $\perp$ to $\mathcal{A}$ if σ turns out to be invalid with respect to (ID_S, ID_V, M). Otherwise, the challenger produces an Aid on behalf of the signer ID_S or the designated verifier ID_V, then forwards Aid to $\mathcal{A}$.
- Hash query. When the involved hash functions are modeled by random oracles, $\mathcal{A}$ also performs adaptive queries to the hash functions. The challenger usually responds by randomly picking an element from the output space of the hash function.

Forgery. $\mathcal{A}$ outputs a signer identity ID_S^*, a verifier identity ID_V^*, a message M^*, and a signature σ^*. $\mathcal{A}$ *succeeds* if the following situations hold:

1. σ^* is valid (as verified by ID_V^*) with respect to ID_S^*, ID_V^* and M^*.
2. $\mathcal{A}$ has not made an Extract query on ID_S^*.
3. σ^* was not returned by a previous Sign query on (ID_S^*, ID_V^*, M^*).

$\mathcal{A}$'s advantage in the above game is defined as

$$Adv_{\mathcal{A}} = Pr[\mathcal{A} \text{ succeeds}]$$

where the probability is taken over all coin tosses made by the challenger and $\mathcal{A}$. We note that the above game captures the notion of *strong unforgeability* first introduced by [1].

Definition 3 (Unforgeability). *An IB-DS scheme is ($\epsilon, t, q_E, q_S, q_{DV}, q_{PV}, q_H$)-unforgeable, if there is no adversary who runs in time at most t, makes at most q_E* Extract *oracle queries, q_S* Sign *oracle queries, q_{DV}* DVerify *oracle queries, q_{PV}* PVerify *oracle queries, and q_H* Hash *oracle queries, and has advantage at least ϵ in the above game.*

Invisibility. Informally, the invisibility property requires that it be (computationally) infeasible for any third party to decide whether a signature was indeed produced by a signer ID_S, designated to ID_V, on message M. Invisibility for digital signatures is studied in two flavors. In [6, 26] this notion is phrased in terms of deciding whether a signature σ corresponds to a message M_0 or M_1, while in several other papers [13, 22], it is defined by the infeasibility of an attacker to distinguish a valid signature σ on M adaptively chosen by the attacker from one randomly drawn from the signature space. Since invisibility (of undeniable signatures) in the latter flavor implies that in the former flavor as shown by [13], in this paper we also consider inviability in the sense of [13, 22]. Formally, we consider the following game between a PPT distinguisher $\mathcal{D}$ and a challenger.

Setup. The challenger runs the `Setup` algorithm of the IB-DS scheme to obtain the public parameters `params` and the master secret. It then gives `params` to $\mathcal{D}$ and keeps the master secret to itself.

Phase 1. $\mathcal{D}$ adaptively makes a number of different queries to the challenger. Each query can be either an Extract query, a Sign query, a DVerify query, a PVerify query, or a Hash query. The challenger responds to these queries in the same way as in the unforgeability game.

Challenge. Once $\mathcal{D}$ decides that Phase 1 is over, it outputs a signer identity ID_S^*, a verifier identity ID_V^*, and a message M^*, and submits them to the challenger. The constraint is that ID_V^* must have not been submitted for the Extract oracle. The challenger then generates a random bit $b \in \{0, 1\}$, if $b = 1$, the challenger produces a signature σ^* on (ID_S^*, ID_V^*, M^*) in the same way as the Sign query. Otherwise, it picks a random σ^* from the signature space. In both cases σ^* is forwarded to $\mathcal{D}$.

Phase 2. $\mathcal{D}$ again adaptively performs several oracle queries as it did in Phase 1, subjected to the following restrictions.
- $\mathcal{D}$ cannot make an Extract query on ID_V^*.
- $\mathcal{D}$ cannot make a DVerify or a PVerify query on $(ID_S^*, ID_V^*, M^*, \sigma^*)$.

Guess. Finally $\mathcal{D}$ outputs a bit $b' \in \{0, 1\}$. $\mathcal{D}$ *succeeds* if $b = b'$.

$\mathcal{D}$'s advantage in the above game is defined as

$$Adv_{\mathcal{D}} = Pr[\mathcal{D} \text{ succeeds}] - \frac{1}{2}$$

where the probability is taken over all coin tosses made by the challenger and $\mathcal{D}$.

Definition 4 (Invisibility). *An IB-DS scheme is* $(\epsilon, t, q_E, q_S, q_{DV}, q_{PV}, q_H)$*-invisible, if there is no adversary who runs in time at most* t*, makes at most* q_E Extract *oracle queries,* q_S Sign *oracle queries,* q_{DV} DVerify *oracle queries,* q_{PV} PVerify *oracle queries, and* q_H Hash *oracle queries, and has advantage at least* ϵ *in the above game.*

4 Proposed Identity-Based Directed Signature Scheme

We now present our IB-DS scheme based on the modified SOK-IBS scheme.

- `Setup`. Given security parameter k, the PKG chooses groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order $q \geq 2^k$ with a bilinear pairing e and a generator P of $\mathbb{G}_1$, as described in Section 2.1. He then selects $x \in_R \mathbb{Z}_q^*$ and computes the public key $P_{pub} = xP$, also picks two cryptographic hash functions $H_1, H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$. The PKG now publishes `params` $= (\mathbb{G}_1, \mathbb{G}_2, q, e, P, P_{pub}, H_1, H_2)$, and keeps x secret as the master secret key.
- `Extract`. Given an identity $ID \in \{0,1\}^*$, the PKG computes $Q_{ID} = H_1(ID) \in \mathbb{G}_1$, then computes the user's private key $d_{ID} = xQ_{ID} \in \mathbb{G}_1$.
- `Sign`. To sign a message M to a designated verifier ID_V, the signer with identity ID_S and private key d_{ID_S} performs the following actions.
  1. Pick $r_1, r_2 \in_R \mathbb{Z}_q^*$.
  2. Compute $U = r_1 P$, $W = r_2 Q_{ID_S}$.
  3. Compute $V = d_{ID_S} + r_1 H$, where $H = H_2(ID_S, ID_V, M, U, e(d_{ID_S}, r_2 Q_{ID_V}))$.

  The signature is $\sigma = (U, W, V)$.
- `DVerify`. Given a purported signature $\sigma = (U, W, V)$ on signer ID_S, verifier ID_V and message M, ID_V verifies it with his private key d_{ID_V} as follows.
  1. Compute $H = H_2(ID_S, ID_V, M, U, e(d_{ID_V}, W))$.
  2. Accept the signature if
  $$e(P, V) = e(P_{pub}, Q_{ID_S}) e(U, H), \tag{1}$$
  reject it otherwise.
- `PVerify`. Given a purported signature $\sigma = (U, W, V)$ on signer ID_S, verifier ID_V and message M, to enable a third party T to verify it, either ID_S or ID_V computes $Aid = e(d_{ID_S}, r_2 Q_{ID_V}) = e(d_{ID_V}, W)$ and sends it to T. The latter then computes $H = H_2(ID_S, ID_V, M, U, Aid)$, and verifies the signature by evaluating (1).

Remark 2. In the ***Sign*** algorithm, the signer has to store all the values r_2 to allow public verification of the signatures later. Thus he has to maintain a table where he writes all the tuples (W, r_2) for all the signatures he has generated. This is an efficiency drawback that we share with the scheme of [26], although the authors of [26] have not mentioned it. Note that since $W = r_2 Q_{ID_S}$, it is unnecessary to store any of (M, ID_V, U, V) in the table.

Remark 3. Our extension to the modified SOK-IBS scheme lies in the pairing operation in the H_2 hash function. It prevents any third party from directly verifying the signature. It also allows us to base the invisibility of our scheme on the DBDH assumption, the detailed proof is given in Theorem 2.

4.1 Correctness

To show correctness of our scheme, we show that the `DVerify` algorithm is consistent with the `Sign` algorithm because

$$\begin{aligned} e(P, V) &= e(P, d_{ID_S} + r_1 H_2(ID_S, ID_V, M, U, e(d_{ID_S}, r_2 Q_{ID_V}))) \\ &= e(P_{pub}, Q_{ID_S}) e(U, H_2(ID_S, ID_V, M, U, e(d_{ID_V}, W))) \\ &= e(P_{pub}, Q_{ID_S}) e(U, H). \end{aligned}$$

Correctness of the `PVerify` algorithm is straightforward.

4.2 Security Analysis

In the following two theorems, we prove that our scheme is existentially unforgeable and invisible, respectively. Proofs of both theorems are developed based on the work of [21].

Theorem 1 (Unforgeability). *If a PPT forger $\mathcal{A}$ has an advantage ϵ in forging a signature of the proposed IB-DS scheme when running in time t and asking q_{H_i} queries to random oracles H_i $(i = 1, 2)$, q_E queries to the* Extract *oracle, q_S queries to the* Sign *oracle, q_{DV} queries to the* DVerify *oracle, and q_{PV} queries to the* PVerify *oracle, then the CDH problem can be solved with probability*

$$\epsilon' \geq (\epsilon - \frac{q_S(2q_{H_2} + q_S - 1)}{2 \cdot 4^k} - \frac{1 + (q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k})(\frac{q_E}{q_E + 1})^{q_E + 1} \frac{1}{q_E},$$

within time

$$t' \leq t + (q_{H_1} + q_{H_2} + q_E + 4q_S)t_m + (q_{H_2} + q_S + (q_{H_2} + q_S + 2)(q_{DV} + q_{PV}))t_{bp} + (q_{H_2} + q_S)t_{mm}.$$

Here t_m is the time to compute a scalar multiplication in $\mathbb{G}_1$, t_{bp} is the time for a pairing operation, and t_{mm} is the time to perform a multi-exponentiation in $\mathbb{G}_1$. And for large q_E,

$$\epsilon' \geq (\epsilon - \frac{q_S(2q_{H_2} + q_S - 1)}{2 \cdot 4^k} - \frac{1 + (q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k}) \frac{1}{e \cdot q_E},$$

where e denotes the base of the natural logarithm.

Proof. Let $\mathcal{B}$ be a PPT attacker against the CDH problem. Given a CDH problem instance $(P, A = aP, B = bP)$, $\mathcal{B}$ uses $\mathcal{A}$ to compute abP by acting as a challenger for $\mathcal{A}$ as follows:

Setup. $\mathcal{B}$ initializes $\mathcal{A}$ with $P_{pub} = A$ as the system's overall public key, therefore the hidden master secret is a.

Simulation of Oracles. $\mathcal{A}$ then starts performing oracle queries. These queries are answered by $\mathcal{B}$ as follows (we assume that, without loss of generality, an identity was first submitted to the H_1 oracle before submitted for other oracle queries):

- H_1 queries: when $\mathcal{A}$ submits an identity ID to the H_1 oracle, $\mathcal{B}$ flips a coin $T \in \{0,1\}$ that yields 0 with probability δ, whose value will be determined later, and 1 with probability $1-\delta$. $\mathcal{B}$ then picks $u \in_R \mathbb{Z}_q^*$ and defines

$$H_1(ID) = \begin{cases} uP & \text{if } T = 0, \\ uB & \text{if } T = 1. \end{cases}$$

  $\mathcal{B}$ then returns $H_1(ID)$ to $\mathcal{A}$ and inserts a tuple (ID, u, T) in a list L_1, which is initially empty, to track how it answered the query.
- Extract queries: when $\mathcal{A}$ submits an identity ID to the Extract oracle, $\mathcal{B}$ recovers the corresponding (ID, u, T) entry from L_1. If $T = 0$, it means that $H_1(ID)$ was previously defined as $uP \in \mathbb{G}_1$, $\mathcal{B}$ computes $d_{ID} = uP_{pub} = uA$ and returns it to $\mathcal{A}$. Otherwise, $\mathcal{B}$ outputs "failure" and halts because it is unable to answer the query legitimately.
- H_2 queries: when $\mathcal{A}$ submits a tuple (ID_S, ID_V, M, U, E) to the H_2 oracle, $\mathcal{B}$ first scans a list L_2, which is initially empty, to see whether an entry for (ID_S, ID_V, M, U, E) exists. If it does exist, the previously defined value is returned. Otherwise, $\mathcal{B}$ picks $v \in_R \mathbb{Z}_q^*$, inserts (ID_S, ID_V, M, U, E, v) into L_2 and returns $vP \in \mathbb{G}_1$ as the hash value to $\mathcal{A}$.
- Sign queries: when $\mathcal{A}$ queries the signing oracle on a message M for a signer ID_S, a verifier ID_V, $\mathcal{B}$ first recovers the previously defined values $Q_{ID_S} = H_1(ID_S)$ and $Q_{ID_V} = H_1(ID_V)$ from L_1. It then chooses $r, t, \nu \in_R \mathbb{Z}_q^*$, computes $W = rP$, $U = \nu P_{pub} = \nu A \in \mathbb{G}_1$ and the bilinear pairing

$$E = e(W, d_{ID_V}) = e(rP_{pub}, Q_{ID_V}).$$

  $\mathcal{B}$ halts and outputs "failure" if L_2 already contains an entry for (ID_S, ID_V, M, U, E). Otherwise $\mathcal{B}$ sets $V = tP_{pub} = tA \in \mathbb{G}_1$, and then defines $H_2(ID_S, ID_V, M, U, E)$ as $\nu^{-1}(tP - Q_{ID_S}) \in \mathbb{G}_1$. The tuple (U, W, V) is returned to $\mathcal{A}$ as the directed signature.
- DVerify queries: $\mathcal{D}$ submits (ID_S, ID_V, M) and $\sigma = (U, W, V)$ to $\mathcal{B}$. The latter first recovers (ID_V, u_V, T_V) from L_1. It then proceeds in either of the following directions:
  - If $T_V = 0$, it computes $d_{ID_V} = u_V P_{pub} = u_V A$ and $E = e(d_{ID_V}, W)$, then recovers $H = H_2(ID_S, ID_V, M, U, E)$ from the L_2 list. If this entry does not exist, $\mathcal{B}$ selects $v \in_R \mathbb{Z}_q^*$ and defines it as vP. $\mathcal{B}$ then verifies equation (1) to check the validity of σ. It returns the verification result, which is either 1 (`valid`) or 0 (`invalid`), to $\mathcal{D}$.
  - If $T_V = 1$, $\mathcal{B}$ works on all the possible entries $H_2(ID_S, ID_V, M, U, E)$ for some E.
    1. For each possible entry $H = H_2(ID_S, ID_V, M, U, E)$ for some E, $\mathcal{B}$ evaluates (1). If the verification result is 1 (`valid`), $\mathcal{B}$ returns 1 (`valid`) to $\mathcal{D}$.
    2. If the above operation does not lead $\mathcal{B}$ to return an answer for $\mathcal{D}$, $\mathcal{B}$ then returns 0 (`invalid`) to $\mathcal{D}$.

- PVerify queries: $\mathcal{D}$ submits (ID_S, ID_V, M) and $\sigma = (U, W, V)$ to $\mathcal{B}$. The latter performs basically the same operation as in the simulation of the DVerify oracle. The only difference is the following: when $\mathcal{B}$ deems σ valid (returns 1 in the DVerify simulation), it returns $Aid = E$ to $\mathcal{D}$; when it deems σ invalid (returns 0 in the DVerify simulation), it returns $\perp$ to $\mathcal{D}$.

It is easy to check that, as long as $\mathcal{B}$ does not fail in a query, its simulation of the H_1, Extract, H_2 and Sign oracles is perfect and indistinguishable from that of a real attack game. Although whether $\mathcal{A}$ is able to verify the output of signing oracle depends on whether it knows d_{ID_V}, $\mathcal{B}$'s simulation method ensures that the output signature is unconditionally valid. It has a probability at most $(q_{H_2} + q_S)/2^k$, which occurs in step 1 of the second case, to return a wrong answer to each DVerify or PVerify query.

Forgery and Output. Eventually, $\mathcal{A}$ outputs a signer identity ID_S^*, a verifier identity ID_V^*, a message M^* and a signature $\sigma^* = (U^*, W^*, V^*)$ on (ID_S^*, ID_V^*, M^*). $\mathcal{B}$ then recovers the triples (ID_S^*, u_S^*, T_S^*) and (ID_V^*, u_V^*, T_V^*) from L_1. If $T_S^* = 0$, then $\mathcal{B}$ outputs "failure" and stops. Otherwise, it goes on and the list L_2 must contain at least one entry $(ID_S^*, ID_V^*, M^*, U^*, E^*, v^*)$ for some E^* and v^* with overwhelming probability (otherwise, $\mathcal{B}$ stops and outputs "failure"). $\mathcal{B}$ now proceeds in either of the following ways, depending on whether it knows the exact value of E^* corresponding to ID_V^* and W^*:

- If $T_V^* = 0$, $\mathcal{B}$ first computes ID_V^*'s private key $d_{ID_V^*} = u_V^* A$, then computes $E^* = e(W^*, d_{ID_V^*})$, and recovers the entry $H = H_2(ID_S^*, ID_V^*, M^*, U^*, E^*, v^*)$ from the L_2 list. $\mathcal{B}$ then knows that
$$e(P, V^*) = e(A, Q_{ID_S^*}) \cdot e(U^*, H)$$
with $H = v^* P \in \mathbb{G}_1$ and $Q_{ID_S^*} = u_S^* B \in \mathbb{G}_1$. Therefore
$$e(P, V^* - v^* U^*) = e(A, u_S^* B)$$
and
$$u_S^{*-1}(V^* - v^* U^*) \tag{2}$$
is the solution to the CDH problem instance $(P, A = aP, B = bP)$.
- Otherwise, $\mathcal{B}$ does not know the exact value of E^*, it works on all the entries $(ID_S^*, ID_V^*, M^*, U^*, E^*, v^*)$ for some E^* and v^* in L_2 as follows:
  1. Compute (2) and let the result be $Z \in \mathbb{G}_1$.
  2. Evaluate $e(A, B) \overset{?}{=} e(Z, P)$. If the equation holds, $\mathcal{B}$ outputs Z as the solution to the CDH instance (P, A, B).

Success Probability. We now analyze $\mathcal{B}$'s success probability. At first, in a signing query, $\mathcal{B}$ has 4^k choices of $(U, E) \in \mathbb{G}_1 \times \mathbb{G}_2$ for each H_2 entry, therefore its probability to fail in a signing query due to a conflict on H_2 is at most
$$Pr_1 = \frac{q_{H_2}}{4^k} + \frac{q_{H_2} + 1}{4^k} + \cdots + \frac{q_{H_2} + q_S - 1}{4^k} = \frac{q_S(2q_{H_2} + q_S - 1)}{2 \cdot 4^k}.$$

Secondly, the probability for $\mathcal{A}$ to output a valid forgery (U^*, W^*, V^*) on (ID_S^*, ID_V^*, M^*) without asking the corresponding H_2 query is at most

$$Pr_2 = \frac{1}{2^k}.$$

Third, its probability to fail in a DVerify or PVerify query (by giving a wrong answer) is at most

$$Pr_3 = \frac{(q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k}.$$

Finally, the probability for $\mathcal{B}$ not to fail in a key extraction query and $\mathcal{A}$ produces its forgery on a "good" identity ID_S^* is $\alpha(\delta) = \delta^{q_E}(1-\delta)$. By an analysis developed in [11] and further exercised in [21, 25], this probability is maximal for $\delta_{max} = \frac{q_E}{q_E+1}$ and

$$Pr_4 = \alpha(\delta_{max}) = (\frac{q_E}{q_E+1})^{q_E+1}\frac{1}{q_E}.$$

For large q_E, $Pr_4 \approx 1/(e \cdot q_E)$.

Therefore, when $\delta = \frac{q_E}{q_E+1}$, $\mathcal{B}$'s success probability in solving the CDH problem is at least

$$\begin{aligned}
&\epsilon(1 - Pr_1)(1 - Pr_2)(1 - Pr_3)Pr_4 \\
>&(\epsilon - Pr_1 - Pr_2 - Pr_3)Pr_4 \\
=&(\epsilon - \frac{q_S(2q_{H_2} + q_S - 1)}{2 \cdot 4^k} - \frac{1 + (q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k})(\frac{q_E}{q_E+1})^{q_E+1}\frac{1}{q_E}.
\end{aligned}$$

And for large q_E, this value is

$$(\epsilon - \frac{q_S(2q_{H_2} + q_S - 1)}{2 \cdot 4^k} - \frac{1 + (q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k})\frac{1}{e \cdot q_E}.$$

$\mathcal{B}$'s Running Time. To respond to either a H_1 query, an Extract query or a H_2 query requires t_m; to respond to a Sign query requires $4t_m + t_{bp} + t_{mm}$. To respond to a DVerify or a PVerify query requires at most $(q_{H_2} + q_S + 2)t_{bp}$. In the final output stage, $\mathcal{B}$'s computation differs in two situations as already discussed:

- If $T_V^* = 0$, then $\mathcal{B}$ knows (can compute) the private key of ID_V^*, and it takes $t_m + t_{bp} + t_{mm}$.
- Otherwise $\mathcal{B}$ needs to test each possible entry in L_2 until it eventually finds the right entry. Regarding $e(A, B)$ as a constant, the computation for each entry is $t_{bp} + t_{mm}$.

Therefore the *worst-case* computational cost for the output stage is that of the second case, which is at most

$$q_{H_2}(t_{bp} + t_{mm}),$$

since $\mathcal{B}$ only needs to check the entries produced by the H_2 queries. It then follows that

$$t' \leq t+(q_{H_1}+q_{H_2}+q_E+4q_S)t_m+(q_{H_2}+q_S+(q_{H_2}+q_S+2)(q_{DV}+q_{PV}))t_{bp}+(q_{H_2}+q_S)t_{mm}. \square$$

Theorem 2 (Invisibility). *If a PPT distinguisher $\mathcal{D}$ has an advantage ϵ in breaking the invisibility of our IB-DS scheme when running in time t and asking q_{H_i} queries to random oracles H_i $(i = 1, 2)$, q_E, q_S, q_{DV}, q_{PV} queries to the* Extract *oracle,* Sign *oracle,* DVerify *oracle and* PVerify *oracle, respectively, then the DBDH problem can be solved with advantage*

$$\epsilon' \geq (\epsilon - \frac{(q_S+1)(2q_{H_2}+q_S)}{2 \cdot 4^k} - \frac{(q_{H_2}+q_S)(q_{DV}+q_{PV})}{2^k})(\frac{q_E}{q_E+1})^{q_E+1}\frac{1}{q_E},$$

within time

$$t' \leq t+(q_{H_1}+q_{H_2}+q_E+4q_S)t_m+(q_S+(q_{H_2}+q_S+2)(q_{DV}+q_{PV}))t_{bp}+q_S t_{mm}.$$

Here t_m, t_{bp} and t_{mm} are of the same meanings as in Theorem 1. And for large q_E,

$$\epsilon' \geq (\epsilon - \frac{(q_S+1)(2q_{H_2}+q_S)}{2 \cdot 4^k} - \frac{(q_{H_2}+q_S)(q_{DV}+q_{PV})}{2^k})\frac{1}{e \cdot q_E},$$

where e denotes the base of the natural logarithm.

Proof. We construct a DBDH problem solver $\mathcal{B}$ using $\mathcal{D}$. Let $\mathcal{B}$ be given a random DBDH problem instance $(P, A = aP, B = bP, C = cP, Z)$. $\mathcal{B}$ simulates a challenger for $\mathcal{D}$ as follows.

Setup. $\mathcal{B}$ initializes $\mathcal{D}$ with $P_{pub} = A = aP$ as the system's overall public key, therefore the hidden master secret is a.

Simulation of Oracles. $\mathcal{D}$ then starts performing oracle queries. These queries are answered by $\mathcal{B}$ in the same way as in the proof of Theorem 1 (we again assume that any oracle query involving an identity is preceded by a H_1 query for the same identity).

Challenge. After $\mathcal{D}$ performs a polynomial number of these oracle queries in **Phase 1**, it chooses a signer identity ID_S^*, a verifier identity ID_V^*, a message M^* and submits them to $\mathcal{B}$. $\mathcal{B}$ then recovers (ID_V^*, u_V^*, T_V^*) from L_1. If $T_V^* = 0$, $\mathcal{B}$ outputs "failure" and terminates the simulation. Otherwise, $\mathcal{B}$ chooses $r, t, \nu \in_R \mathbb{Z}_q^*$ and sets $U^* = \nu P_{pub} = \nu A \in \mathbb{G}_1$, $W^* = rC$, $V^* = tP_{pub} = tA \in \mathbb{G}_1$. Since $H(ID_V^*)$ was previously defined as $u_V^* B$, it follows that

$$e(d_{ID_V^*}, W^*) = e(a \cdot u_V^* B, rC) = e(P, P)^{abc(r \cdot u_V^*)}.$$

Accordingly, to implant the DBDH problem into the challenge signature, $\mathcal{B}$ then inserts

$$H_2(ID_S^*, ID_V^*, M^*, U^*, Z^{r \cdot u_V^*}) \Longrightarrow \nu^{-1}(tP - Q_{ID_S^*}) \qquad (3)$$

into the L_2 list, and forwards $\sigma^* = (U^*, W^*, V^*)$ to $\mathcal{D}$ as the challenge signature. If an entry was already defined for the input $(ID_S^*, ID_V^*, M^*, U^*, Z^{r \cdot u_V^*})$, $\mathcal{B}$ outputs "failure" and halts.

U^*, W^* and V^* in the challenge signature are uniformly random and pairwise independent, therefore whether σ^* is a valid signature on (ID_S^*, ID_V^*, M^*) depends on the H_2 entry defined by (3). It is not hard to see that, if $(P, A = aP, B = bP, C = cP, Z)$ is a real BDH tuple, the H_2 entry produced in (3) is a 'real' hash entry for (ID_S^*, ID_V^*, M^*) and σ^*, then σ^* is a real valid signature and of the same distribution as that in a actual game. Otherwise, the H_2 entry produced in (3) does not corresponds to (ID_S^*, ID_V^*, M^*) and σ^*, as $Z^{r \cdot u_V^*}$ is unrelated to $e(d_{ID_V^*}, W^*)$, then σ^* is not related to (ID_S^*, ID_V^*, M^*) in a meaningful way, hence randomly chosen from the signature space in the viewpoint of $\mathcal{D}$.

Therefore $\mathcal{B}$'s simulation of the challenge signature is indistinguishable from a real game, as long as it does not fail. And the challenge bit b, which is hidden from $\mathcal{B}$, is actually

$$b = \begin{cases} 1 & \text{if } Z = (P, P)^{abc}, \\ 0 & \text{otherwise.} \end{cases}$$

Guess and Output. After performing several oracle queries in **Phase 2** subjected to the proper restrictions, $\mathcal{D}$ outputs a bit $b' \in \{0, 1\}$ as its guess of b. $\mathcal{B}$ then outputs b' as the answer to the DBDH problem instance $(P, A = aP, B = bP, C = cP, Z)$. This is justified by the above analysis on the challenge signature and the hidden challenge bit.

$\mathcal{B}$'s Advantage. Similar to Theorem 1, $\mathcal{B}$'s probability to fail in a signing query or the **Challenge** stage due to a conflict on H_2 is at most

$$Pr_1 = \frac{(q_S + 1)(2q_{H_2} + q_S)}{2 \cdot 4^k}.$$

Its probability to fail in a DVerify or PVerify query (by giving a wrong answer) is at most

$$Pr_2 = \frac{(q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k}.$$

The probability for $\mathcal{B}$ not to fail in a key extraction query, and $\mathcal{D}$ requests challenge on a "good" identity ID_V^* is $\alpha(\delta) = \delta^{q_E}(1 - \delta)$. This probability is maximal for $\delta_{max} = \frac{q_E}{q_E+1}$ and

$$Pr_3 = \alpha(\delta_{max}) = (\frac{q_E}{q_E + 1})^{q_E+1} \frac{1}{q_E}.$$

For large q_E, $Pr_3 \approx 1/(e \cdot q_E)$.

Eventually, it follows that, when $\delta = \frac{q_E}{q_E+1}$, $\mathcal{B}$'s advantage in solving the DBDH problem is at least

$$\begin{aligned} &(\epsilon - Pr_1 - Pr_2)Pr_3 \\ =&(\epsilon - \frac{(q_S + 1)(2q_{H_2} + q_S)}{2 \cdot 4^k} - \frac{(q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k})(\frac{q_E}{q_E + 1})^{q_E+1} \frac{1}{q_E}. \end{aligned}$$

And for large q_E, this value is

$$(\epsilon - \frac{(q_S + 1)(2q_{H_2} + q_S)}{2 \cdot 4^k} - \frac{(q_{H_2} + q_S)(q_{DV} + q_{PV})}{2^k}) \frac{1}{e \cdot q_E}.$$

Table 1. Comparison of our scheme to previous schemes

Scheme		SOK-IBSE [21]	Wang [28]	Our Scheme
Signature/Ciphertext Size		$2\lvert\mathbb{G}_1\rvert + l$	$\lvert\mathbb{G}_1\rvert + \lvert\mathbb{Z}_q^*\rvert$	$3\lvert\mathbb{G}_1\rvert$
Formal Model and Proof		Yes	No	Yes
Computation Costs	Signing	$4t_m + t_{bp} + t_{\mathcal{E}}$	$3t_m + t_{bp}$	$4t_m + t_{bp}$
	Direct Verify	$t_m + 4t_{bp} + t_{\mathcal{D}}$	$t_m + 2t_{bp}$	$4t_{bp}$
	Public Verify	$3t_{bp}$	Unsupported	$3t_{bp}$

$\mathcal{B}$'s Running Time. The computational cost for answering each oracle query is same as that of Theorem 1. In assessing $\mathcal{B}$'s total running time, if we omit the cost in the **Challenge** stage (which is $3t_m + 1$ exponentiation in $\mathbb{G}_2 + t_{mm}$), we have

$$t' \leq t + (q_{H_1} + q_{H_2} + q_E + 4q_S)t_m + (q_S + (q_{H_2} + q_S + 2)(q_{DV} + q_{PV}))t_{bp} + q_S t_{mm}. \square$$

We have shown that our proposed IB-DS scheme is unforgeable and invisible assuming intractability of the CDH problem and the DBDH problem in $\mathbb{G}_1$, respectively.

5 Efficiency and Comparison

We now analyze the efficiency of our proposed scheme and compare it with the related schemes, including the SOK-IBSE scheme [21] and the IB-DS scheme of Wang [28]. The comparison is summarized in Table 1. Here l is the ciphertext length of the symmetric cipher $(\mathcal{E}, \mathcal{D})$ used by the SOK-IBSE scheme, $t_{\mathcal{E}}$ is the computational cost to perform encryption of the scheme, and $t_{\mathcal{D}}$ for decryption. Note that we don't consider hash function evaluation, point addition in $\mathbb{G}_1$ and multiplication in $\mathbb{G}_2$ as they are much cheaper than other operations.

As shown by Table 1, our scheme has a comparable signature/ciphertext size with the SOK-IBSE scheme, and it is more efficient than the latter scheme in signing and direct verification, due to the fact that the latter performs the additional message encryption. Public verification in the two schemes has the same computation cost because they are both based on the SOK-IBS scheme.

Compared with Wang's scheme, our scheme is slightly less efficient. However it offers more security guarantee than Wang's scheme by admitting rigorous security proofs under a reasonable security model, and provides the feature of public verification.

6 Conclusions

Directed signatures are applicable where the designated receiver needs to exclusively verify a signature, and shares, with the signer, the ability to prove validity of the signature to others. In this paper, we studied directed signatures in the identity-based setting. We first formally defined the syntax and security notions for identity-based directed signatures (IB-DS), then proposed a concrete IB-DS scheme from bilinear pairings. In the random oracle model, our scheme is proved to be unforgeable and invisible based on the intractability of CDH problem and DBDH problem, respectively. In the future, we are interested in designing an identity-based directed signature scheme without relying on the (expensive) bilinear pairings.

References

1. J.H. An, Y. Dodis, T. Rabin, On the security of joint signature and encryption, Advances in Cryptology - EUROCRYPT'02, LNCS 2332, Springer-Verlag, 2002, pp. 83–107.
2. S. Araki, S. Uehara, K. Imamura, The limited verifier signature and its application, IEICE Trans Fundamentals E82-A (1) (1999) 63–68.
3. M. Bellare, C. Namprempre, G. Neven, Security proofs for identity-based identification and signature schemes, Advances in Cryptology - EUROCRYPT'04, LNCS 3027, Springer-Verlag, 2004, pp. 268–286.
4. M. Bellare, P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, Proceedings of the First Annual Conference on Computer and Commmunications Security, ACM Press, 1993, pp. 62–73.
5. D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing, Advances in Cryptology - CRYPTO'01, LNCS 2139, Springer-Verlag, 2001, pp. 213-229.
6. J. Camenisch, M. Michels, Confirmer signature schemes secure against adaptive adversaries, Advances in Cryptology - EUROCRYPT'00, LNCS 1870, Springer-Verlag, 2000, pp. 243–258.
7. J.C. Cha, J.H. Cheon, An identity-based signature from gap Diffie-Hellman groups, Public Key Cryptography - PKC'03, LNCS 2567, Springer-Verlag, 2003, pp. 18–30.
8. D. Chaum, H. van Antwerpen, Undeniable signatures, Advances in Cryptology - CRYPTO'89, LNCS 435, Springer-Verlag, 1989, pp. 212–216.
9. D. Chaum, Designated confirmer signatures, Advances in Cryptology - EUROCRYPT'94, LNCS 950, Springer-Verlag, 1994, pp. 86–91.
10. X. Chen, F. Zhang, K. Kim, Limited verifier signature from bilinear pairings, Proceedins of Applied Cryptography and Network Security 2004, LNCS 3089, Springer-Verlag, 2004, pp. 135–148.
11. J.S. Coron, On the exact security of Full Domain Hash, Advances in Cryptology - CRYPTO'00, LNCS 1880, Springer-Verlag, 2000, pp. 229–235.
12. R. Dutta, R. Barua, P. Sarkar, Pairing-based cryptographic protocols: A survey. Cryptology ePrint Archive, 2004.
13. S. Galbraith, W. Mao, Invisibility and anonymity of undeniable and confirmer signatures, CT-RSA 2003, LNCS 2612, Springer-Verlag, 2003, pp. 80–97.
14. D. Galindo, J. Herranz, E. Kiltz, On the generic construction of identity-based signatures with additional properties, Advances in Cryptology - ASIACRYPT'06, LNCS 4284, Springer-Verlag, 2006, pp. 178–193.
15. S. Goldwasser, S. Micali, R.L. Rivest, A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing 17(2) (1988) 281–308.
16. L.C. Guillou, J.J. Quisquater, A paradoxical identity-based signature scheme resulting from zero-knowledge, Advances in Cryptology - CRYPTO'88, LNCS 403, Springer-Verlag, 1988, pp. 216–231.

17. F. Hess, Efficient identity based signature schemes based on pairings, SAC 2002, LNCS 2595, Springer-Verlag, 2003, pp. 310–324.
18. M. Jakobsson, K. Sako, R. Impagliazzo, Designated verifier proofs and their applications, Advances in Cryptology - EUROCRYPT'96, LNCS 1070, Springer-Verlag, 1996, pp. 143–154.
19. F. Laguillaumie, P. Paillier, D. Vergnaud, Universally convertible directed signatures, Advances in Cryptology - ASIACRYPT'05, LNCS 3788, Springer-Verlag, 2005, pp. 682–701.
20. S. Lal, M. Kumar, A directed signature scheme and its applications, arXiv:cs/0409036, 2004.
21. B. Libert, J.J. Quisquater, The exact security of an identity based signature and its applications. http://eprint.iacr.org/2004/102.
22. B. Libert, J.J. Quisquater, Identity based undeniable signatures, CT-RSA 2004, LNCS 2964, Springer-Verlag, 2004, pp. 112–125.
23. C.H. Lim, P.J. Lee, Modified Maurer-Yacobi's scheme and its applications, Advances in Cryptology - AUSCRYPT'92, LNCS 718, Springer-Verlag, 1992, pp. 308–323.
24. Y. Long, K. Chen, Certificateless threshold cryptosystem secure against chosen-ciphertext attack, Information Sciences 177 (24) (2007) 5620–5637.
25. R. Lu, Z. Cao, A directed signature scheme based on RSA assumption, International Journal of Network Security 2 (3) (2006) 182–421.
26. R. Lu, X. Lin, Z. Cao, J. Shao, X. Liang, New (t, n) threshold directed signature scheme with provable security, Information Sciences 178 (3) (2008) 756–765.
27. A. Shamir, Identity based cryptosystems and signature schemes, Advances in Cryptology - CRYPTO'84, LNCS 196, Springer-Verlag, 1984, pp. 47–53.
28. Y. Wang, Directed signature based on identity, Journal of Yulin College 15 (5) (2005) 1–3. In Chinese.
29. L.C. Wang, Z.F. Cao, X.X. Li, H.F. Qian, Simulatability and security of certificateless threshold signatures, Information Sciences 177 (6) (2007) 1382–1394.
30. J. Zhang, J. Mao, A novel ID-based designated verifier signature scheme, Information Sciences 178 (3) (2008) 766–773.
31. F. Zhang, K. Kim, A universal forgery of Araki *et al.*'s convertible limited verifier signature scheme, IEICE Trans. Fundamentals E86-A (2) (2003) 515–516.
32. Y. Zheng, Digital signcryption or how to achieve cost(signature & encryption)<< cost(signature)+cost(encryption), in: Advances in Cryptology-CRYPTO'97, LNCS, vol. 1294, Springer-Verlag, 1997, pp. 165– 179.