INFORMATICA, 2009, Vol. 20, No. 4, 461–476

One-Round ID-Based Threshold Signature Scheme from Bilinear Pairings *

Wei GAO[1,2], Guilin WANG[3], Xueli WANG[4], Zhenguang YANG[1]

[1]*School of Mathematics & Information, Ludong University*
Yantai 264025, P.R. China
[2]*Guangdong Key Lab of Information Security Technology, Sun Yat-sen University*
Guangzhou 510275, P.R. China
[3]*School of Computer Science, University of Birmingham*
Birmingham B15 2TT, UK
[4]*School of Mathematics, South China Normal University*
Guangzhou 510631, P.R. China
e-mail: sdgaowei@gmail.com; maths@ldu.edu.cn; sdgaowei@yahoo.com.cn;
g.wang@cs.bham.ac.uk; wangxuyuyan@yahoo.com.cn

Received: December 2007; accepted: June 2008

Abstract. In this paper, we propose a new ID-based threshold signature scheme from the bilinear pairings, which is provably secure in the random oracle model under the bilinear Diffie–Hellman assumption. Our scheme adopts the approach that the private key associated with an identity rather than the master key of PKG is shared. Comparing to the-state-of-art work by Baek and Zheng, our scheme has the following advantages. (1) The round-complexity of the threshold signing protocol is optimal. Namely, during the signing procedure, each party broadcasts only one message. (2) The communication channel is optimal. Namely, during the threshold signing procedure, the broadcast channel among signers is enough. No private channel between any two signing parties is needed. (3) Our scheme is much more efficient than the Baek and Zheng scheme in term of computation, since we try our best to avoid using bilinear pairings. Indeed, the private key of an identity is indirectly distributed by sharing a number $x_{ID} \in \mathbb{Z}_q^*$, which is much more efficient than directly sharing the element in the bilinear group. And the major computationally expensive operation called distributed key generation protocol based on the bilinear map is avoided. (4) At last, the proactive security can be easily added to our scheme.

Keywords: identity-based signature, threshold signature, bilinear pairing.

1. Introduction

In 1984, Shamir asked for ID-based encryption and signature schemes to simplify key management procedures in certificate-based public key setting. Since then, many ID-based cryptographic schemes (Boneh and Franklin, 2001; Bellare *et al.*, 2004; Kancaharl *et al.*, 2005; Qian *et al.*, 2005) have been proposed. The ID-based public key setting can

*This work is partially supported by National Natural Science Foundation of China CNF10771078 and Open Fund of Guangdong Key Laboratory of Information Security Technology.

be an alternative for certificate-based public key setting, especially when efficient key management and moderate security are required.

Secret key exposure due to non-cryptographic reasons, such as the underlying machine or system compromise, human errors, and insider attacks, may be the greatest practical threat to many cryptographic protocols. The most common remedy is to distribute secret information (i.e., a secret key) and computation (i.e., signature generation or decryption) among n parties. The goal is to allow any subset of more than t parties to jointly reconstruct a secret and/or perform the computation, while preserving security even in the presence of a malicious adversary which can corrupt up to t (the threshold) parties. A review of research on threshold cryptography under CA-based public key setting was presented in Desmedt (1994).

Combining the above two concepts to realize "ID-based threshold signature" is the focus of this paper. To the best of our knowledge, there exist two ID-based threshold signature schemes (Baek and Zheng, 2004; Chen *et al.*, 2004) which adopt the approach that the private key associated with an identity rather than the master key of PKG is shared (Boneh and Franklin, 2001). In Baek and Zheng (2004) the authors first proposed a suit of secret sharing schemes based bilinear pairings, such as Computationally secure Verifiable Secret-Sharing scheme based on the Bilinear Map (CVSSBM) and Distributed Key generation Protocol Based on the Bilinear Map (DKPBM). Then they constructed the first ID-based threshold signature scheme by applying these basic tools to the Hess's ID-based signature scheme (Hess, 2002). We observe one sharp difference between these two schemes: Chen *et al.*'s scheme improperly reduces the distributed key generation protocol which, however, is the most expensive operation in the Baek–Zheng scheme. This improper reduction results in the greatly weakened robustness property: the claimed robustness in Chen *et al.* (2004) means no more than that (1) the validity of any signature share can be publicly verified, and (2) the threshold signing protocol completes successfully only if all signature shares is valid. In contrast, the claimed robustness in Baek and Zheng (2004) strictly follows the standard notion (see Definition 5 in Section 3).

Our main contribution is that we propose a new ID-based threshold signature scheme from the bilinear pairings, which is provably secure in the random oracle model (Bellare and Rogaway, 1993) under the bilinear Diffie–Hellman assumption. On one hand, our scheme shares with the Baek–Zheng scheme some advantages such as optimal-resilience and the approach that the private key associated with an identity rather than the master key of PKG is shared. On the other hand, a comparison with the Baek–Zheng scheme shows the following advantages due to our scheme. (1) The round-complexity of the threshold signing protocol is optimal. Namely, during the signing procedure, each party broadcasts only one message. (2) The communication channel is optimal. Namely, during the threshold signing procedure, the broadcast channel is enough. No private channel between any two signing parties is needed. (3) Our scheme is much more efficient than the Baek and Zheng scheme in term of computation, since we try our best to avoid using bilinear pairings. Indeed, the private key of an identity is indirectly distributed by sharing a number $x_{ID} \in \mathbb{Z}_q^*$, which is much efficient than directly sharing the element in the bilinear group as in Baek and Zheng (2004). And the major computationally expensive

protocol called distributed key generation protocol based on the bilinear map (Baek and Zheng, 2004) is avoided. (4) At last, the proactive security can be easily added to our scheme.

2. Bilinear Pairings and BDH Assumption

In this section, we present the definitions of bilinear pairings and the bilinear Diffie–Hellman (BDH) assumption.

DEFINITION 1. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of prime order q and let P be a generator of $\mathbb{G}_1$. The map e: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is said to be a bilinear pairing if the following three conditions hold: (i) e is bilinear, i.e., $e(aP, bP) = e(P, P)^{ab}$ for all $a, b \in \mathbb{Z}_q^*$; (ii) e is non-degenerate, i.e., $e(P, P) \neq 1$; (iii) e is efficiently computable. Such a group $\mathbb{G}_1$ is called a bilinear group.

DEFINITION 2. Let e: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of prime order q and P is a generator of $\mathbb{G}_1$. We define the bilinear Diffie–Hellman (BDH, for short) problem with respect to $(\mathbb{G}_1, \mathbb{G}_2, e, P)$ as following: given random $aP, bP, cP \in \mathbb{G}_1$, output $e(P, P)^{abc}$. Accordingly, the bilinear Diffie–Hellman assumption states that: there is no PPT algorithm with non-negligible probability which can solve the BDH problem.

3. Models and Formal Frameworks

In this section we introduce the communication model and the adversary model, and then provide definitions of secure ID-based threshold signature schemes. For more details, we refer readers to Baek and Zheng (2004).

3.1. *Communication Model and Adversary Model*

- *Communication Model.* We assume that the involved n signing parties are connected by a broadcast channel. Furthermore, any one pair of the signing parties is connected by a private channel. And there is a private channel between the dealer and any signing participant. However, in this paper, the private channel between any two signing parties is not required.
- *Adversary Model.* We consider an adversary who chooses corrupted signing parties at the beginning of each time period and may cause corrupted players to divert from the protocol in any way. Such an adversary is said to be static and malicious.

3.2. *Framework of ID-Based Threshold Signature*

DEFINITION 3. An identity-based signatures scheme $\mathcal{IDS}$ is a collection of the following four algorithms:

- Setup. This algorithm is run by the trusted party called PKG on input a security parameter, and generates the public parameters $params$ of the scheme and a master secret. PKG publishes $params$ and keeps the master secret to itself.
- Extract. Given an identity ID, the master secret and $params$, this algorithm generates the private key D_{ID} of ID.
- Sign. Given a message m, an identity ID, a private key D_{ID} and $params$, this algorithm generates the signature σ of ID on m.
- Verify. Given a signature σ, a message m, an identity ID and $params$, this algorithm outputs 1 if σ is a valid signature on m for identity ID, or 0 otherwise.

In addition to the above algorithms, an ID-based (t, n)-threshold signature scheme $\mathcal{IDTS}$ based on the $\mathcal{IDS}$ has two additional algorithms as follows:

- ShareKey. Given a private key D_{ID} associated with an identity ID, the size of signature generation players n and the threshold parameter t ($1 < t < n$), this algorithm generates a share $D_{ID,i}$ of D_{ID} for each player P_i, $i = 1, 2, \ldots, n$. At least $t + 1$ shares can be used to reconstruct D_{ID}. Sometimes, it also publishes some information that can be used to check the validity of each share of the private key.
- TSign. Given the common parameter $params$ and a message m, n players, each of them (P_i) holding a share $D_{ID,i}$ of D_{ID}, jointly generate a signature σ of ID on m.

Besides the natural requirement on correctness, i.e., the signature generated by more than t signers must be valid, the security definition for an ID-based threshold signature scheme includes unforgeability and robustness (Baek and Zheng, 2004).

DEFINITION 4 (unforgeability). Let $\mathcal{A}$ be an attacker assumed to be a probabilistic Turing machine taking a security parameter k as input. Consider the following game in which $\mathcal{A}$ interacts with the "Challenger" $\mathcal{C}$:

- *Phase* 1. $\mathcal{A}$ corrupts t players which, without loss of generality, are assumed to be $P_1, \ldots, P_t$. (That is, the attacker is assumed to be static; Gennaro *et al.*, 2001.)
- *Phase* 2. When $\mathcal{A}$ issues a private key extraction query on an identity ID, $\mathcal{C}$ runs Extract taking ID as input and returns the private key D_{ID} to $\mathcal{A}$.
- *Phase* 3. When $\mathcal{A}$ submits a target identity ID^*, $\mathcal{C}$ runs Extract to get a private key D_{ID^*} for ID^*. Subsequently, it runs ShareKey to share D_{ID^*} among $P_1, \ldots, P_n$.
- *Phase* 4. When $\mathcal{A}$ issues a signature generation query for a message m, $\mathcal{C}$ plays the role of the uncorrupted players, in the execution of TSign to collectively generate the signature σ on m. Note that in this phase, $\mathcal{A}$ is allowed to issue private key extraction queries (identities) except for ID^*.
- *Phase* 5. $\mathcal{A}$ outputs (ID^*, m^*, σ^*), where σ^* is a signature with respect to the message m^* and the identity ID^*. A restriction here is that $\mathcal{A}$ must not make a private key extraction query for ID^* or a signature generation query for m^*.

We define $\mathcal{A}$'s success by

$$Adv_{\mathcal{IDTS},\mathcal{A}}^{\mathrm{UF-IDTS-CMA}}(k) = \Pr\left[\mathsf{Verify}(\sigma^*, m^*, ID^*, params) = 1\right].$$

The ID-based threshold signature scheme $\mathcal{IDTS}$ is said to be UF–IDTS–CMA secure if for any polynomial-time attacker $\mathcal{A}$, $Adv_{\mathcal{IDTS},\mathcal{A}}^{\mathrm{UF-IDTS-CMA}}(k)$ is negligible in k.

DEFINITION 5 (robustness). An ID-based (t, n)-threshold signature scheme is (t', n)-robust if it computes a correct output even in the presence of a malicious attacker that makes t' corrupted players deviate from the normal execution.

Motivated by Gennaro *et al.*'s (2001) methodology for proving the security of threshold signature, Baek and Zheng (2004) defined simulatability of $\mathcal{IDTS}$. The following definition, which is a slight extension of that in Baek and Zheng (2004), is actually a stronger property than the above definition of unforgeability.

DEFINITION 6. An ID-based threshold signature scheme $\mathcal{IDTS}$ is simulable if the following properties hold:

- The protocol ShareKey is simulatable. That is, there exists a simulator SIM_1 that, on input the public parameter $params$ and the identity ID, can simulate the view of the attacker on an execution of ShareKey of $\mathcal{IDTS}$ for distributing the private key D_{ID}.
- The protocol TSign is simulatable. That is, there exists a simulator SIM_2 that, on input the public parameter $params$, the identity ID and the message m, t shares $D_{ID,1}, \ldots, D_{ID,t}$, and the public outputs (including the signature σ and other public information) of TSign of $\mathcal{IDTS}$, can simulate the view of the adversary on an execution of TSign for generating a signature σ' of ID on the message m.
  Note that σ' is not required to be the same to the input σ while in Baek and Zheng (2004), Gennaro *et al.*'s (2001) it is required that $\sigma' = \sigma$.

In the full paper of Baek and Zheng (2004) the authors indeed proved that if the underlying ID-based signature scheme is unforgeable and the ID-based threshold signature is simulatable, then the ID-based threshold signature scheme is unforgeable.

4. Construction

4.1. *ID-Based Threshold Signature Scheme*

First, we present the construction of the underlying ID-based signature scheme $\mathcal{IDS}$, which is a tuple of (Setup, Extract, Sign, Verify) as follows:

- Setup. The Private Key Generator (PKG) generates the public parameters and master secret as follows:
  1) generates groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order q with bilinear pairing e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$;
  2) chooses an arbitrary generator $P \in \mathbb{G}_1$;
  3) picks a random $s \in \mathbb{Z}_q^*$ and sets $P_{\mathrm{pub}} = sP$;

4) chooses cryptographic hash functions H_1, H_2: $\{0,1\}^* \rightarrow \mathbb{G}_1$. The PKG's public parameter is $params = (\mathbb{G}_1, \mathbb{G}_2, e, q, P, P_{\rm pub}, H_1, H_2)$; its master secret is $s \in \mathbb{Z}_q^*$.

- Extract. The signer with identity ID receives the value $D_{ID} = sQ_{ID}$ from the PKG as its private key, where $Q_{ID} = H_1(ID) \in \mathbb{G}_1$. Additionally, the signer with identity ID himself selects a random number $x_{ID} \in \mathbb{Z}_q^*$ and computes $B = x_{ID}P$, $C = x_{ID}Q_{ID} + D_{ID}$.
- Sign. For the given message m, the signer with the identity ID first computes $P_m = H_2(ID, m)$ and then sets the signature $\sigma \leftarrow (A, B, C)$, where $A = x_{ID}P_m$.
- Verify. Let (A, B, C) be the signature on the message m with respect to ID and $P_m = H_2(ID, m)$. The verifier checks that

$$e(A, P) = e(P_m, B), \qquad e(Q_{ID}, P_{\rm pub} + B) = e(C, P).$$

We now give some remarks on our $\mathcal{IDS}$. First, as shown in Theorem 1 (see later in Section 5), our $\mathcal{IDS}$ is *existential forgery* under an adaptive chosen ID and message attack in the random oracle model. Nevertheless, it is not *strong* existential forgery, since given a valid signature (A, B, C) on a message m, one can trivially derive a new signature (A', B', C') for the same message m as follows: By selecting a random x, set $A' = A + xP_m$, $B' = B + xP$, and $C' = C + xQ_{ID}$. Second, in usual cases our $\mathcal{IDS}$ is less efficient than the existing schemes in Hess (2002): (a) The signature in our $\mathcal{IDS}$ consists of three elements, while both schemes in Hess (2002) need two elements; and (b) To generate and verify a signature, our $\mathcal{IDS}$ requires to carry out four bilinear mapping, while this number is two in Hess (2002). However, our $\mathcal{IDS}$ will perform better and has comparable efficiency as the schemes (Hess, 2002) in the scenarios, where the signer needs to issue multiple signatures to the same recipient. The reason is that in such cases, B and C can be transferred and verified just for one time (not multiple times!). Actually, this property may have independent interest. Most importantly, as our main contribution, we notice that our $\mathcal{IDS}$ can be used to construct efficient threshold signature scheme, while it seems that there exist no analogous threshold solutions for $\mathcal{IDS}$ in Hess (2002). The reason is that in our $\mathcal{IDS}$, we can simply use Feldman's verifiable threshold secret sharing scheme (Feldman, 1987) to distribute the secret key x_{ID}, and during the threshold signing procedure, our scheme does not require n players to generate a random element in $\mathbb{G}_1$ collectively, as shown below.

To construct the corresponding ID-based threshold signature scheme $\mathcal{IDTS}$ for the above $\mathcal{IDS}$, we design the following two algorithms:

- KeyShare. The private key D_{ID} of the dealer with the identity ID can be distributed among n parties $P_1, P_2, \ldots, P_n$ as follows:
  1. The dealer selects a random number $x_{ID} \in \mathbb{Z}_q^*$ and broadcasts $B = x_{ID}P$, $C = x_{ID}Q_{ID} + D_{ID}$. To check the validity of the public information (B, C), each party P_i checks the equation $e(Q_{ID}, P_{\rm pub} + B) = e(C, P)$.
  2. The dealer distributes $x_{ID} \in \mathbb{Z}_q^*$ using the well-known verifiable (t, n)-threshold secret sharing scheme due to Feldman (1987) as follows. First, the

dealer selects a random polynomial $f(x) = a_0 + a_1 x + \cdots + a_t x^t \in \mathbb{Z}_q^*[x]$ such that $f(0) = a_0 = x_{ID}$, while other a_i's are random numbers selected from $\mathbb{Z}_q$. Next, the dealer broadcasts $a_0 P, a_1 P, \ldots, a_t P$. Then, the dealer secretly sends each party P_i the private key share $x_{ID,i} = f(i) \in \mathbb{Z}_q^*$. To verify the validity of his share $(x_{ID,i})$, each party P_i checks the equation

$$x_{ID,i} P = \sum_{j=0}^{t} i^j (a_j P).$$

Here note that D_{ID} is not directly but indirectly distributed to the n parties: the shared x_{ID} can be used to immediately derive the private key $D_{ID} = C - x_{ID} Q_{ID}$.

- TSign. Let m be an message to be signed. The n players $P_1, \ldots, P_n$ jointly generate the signature as follows.
  1. Each Party P_i generates his partial signature $A_i = x_{ID,i} P_m$ and then broadcasts A_i, where $P_m = H_2(ID, m) \in \mathbb{G}_1$. The validity of the signature share A_i due to player P_i can be publicly verified by checking

$$e(A_i, P) = e\left(H_2(ID, m), \sum_{j=0}^{t} i^j (a_j P)\right).$$

  2. Each party locally reconstructs the full signature as follows. He first finds $t + 1$ valid signature shares using the above verification equation. Suppose that Φ is the set of indices of $t + 1$ honest players who generated valid signature shares. The resulting signature is set as $\sigma = (A, B, C)$ by computing $A = \sum_{i \in \Phi} L_{\Phi,i}(0) A_i$, where $L_{\Phi,i}(0)$ is the appropriate Lagrange coefficient such that $x_{ID} = \sum_{i \in \Phi} L_{\Phi,i}(0) x_{ID,i}$. It is easy to see the correctness of this threshold signature scheme. In fact

$$\begin{aligned} A &= \sum_{i \in \Phi} L_{\Phi,i}(0) A_i = \sum_{i \in \Phi} L_{\Phi,i}(0) \sum_{j=0}^{t} (a_j i^j) H_2(ID, m) \\ &= \sum_{i \in \Phi} L_{\Phi,i}(0) f(i) H_2(ID, m) = x_{ID} H_2(ID, m). \end{aligned}$$

4.2. *Adding Proactive Security*

The idea of the proactive approach is to periodically renew shares of a secret such that information gained by an adversary learning some number of shares (less than a threshold) in one time period will be useless for the adversary's next attacks in the future time periods when all shares are renewed. Proactive secret sharing algorithm PSS has been proposed in Herzberg *et al.* (1995). In Herzberg *et al.* (1995), the authors prove that the security of the robust threshold signature scheme will be preserved when used with PSS protocol if it is a discrete log based robust threshold signature scheme, in which

threshold key generation protocol implements Shamir's secret sharing of the secret signature key x corresponding to the public key $y = g^x$ and outputs verification information $(g^{x_1}, \ldots, g^{x_n})$, where $(x_1, \ldots, x_n)$ are secret shares of the players and if the threshold signature protocol is simulatable. As it is easy to note that our ID-based threshold signature scheme meets all these requirements. Thus the above ID-based threshold signature scheme can be proactivized using PSS and methods of Herzberg *et al.* (1995).

5. Security Analysis

In this section, we first prove the security of the underlying ID-based signature scheme. Then we prove that the two algorithms KeyShare, TSign is simulatable. At last, the security of the proposed ID-based threshold signature scheme is immediately derived. For the formal definition of ID-based signature scheme, we refer the readers to Cha and Cheon (2003).

Theorem 1. *If there is no t'-time algorithm which can solve the bilinear Diffie–Hellman problem on $(\mathbb{G}_1, \mathbb{G}_2)$ with probability at least ϵ', then the ID-based signature scheme on $(\mathbb{G}_1, \mathbb{G}_2)$ is $(t, q_E, q_S, q_{H_1}, q_{H_2}, \epsilon)$-secure against existential forgery under an adaptive chosen ID and message attack in the random oracle model, where the adversary's execution time t and advantage ϵ satisfying*

$$\epsilon \geqslant \mathrm{e}^2(q_S + 1)(q_E + 1)\epsilon', \qquad t \leqslant t' - 4c_{\mathbb{G}_1}(q_{H_1} + q_{H_2} + 3q_S + 2q_E).$$

Here $c_{\mathbb{G}_1}$ is a constant that depends on $\mathbb{G}_1$, e is the base of the natural logarithm, and $q_E, q_S, q_{H_1}, q_{H_2}$ are respectively the numbers of queries that the forger $\mathcal{F}$ can ask to the extract oracle, the signing oracle, the H_1 oracle and the H_2 oracle.

Proof. Suppose that $\mathcal{F}$ is a forger algorithm that $(t, q_E, q_S, q_{H_1}, q_{H_2}, \epsilon)$-breaks the proposed ID-based signature scheme. We will show how to construct a t'-time algorithm $\mathcal{A}$ that solves the Bilinear Diffie–Hellman problem with probability at least ϵ'.

Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of prime order q, P be a generator of $\mathbb{G}_1$ and e: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing. $\mathcal{A}$ is given $X, Y, Z \in \mathbb{G}_1$, where $X = xP, Y = yP, Z = zP \in \mathbb{G}_1, x, y, z \in_R \mathbb{Z}_q^*$. $\mathcal{A}$'s goal is to compute $e(P, P)^{xyz}$. $\mathcal{A}$ simulates the challenger and interacts with forger $\mathcal{F}$ as follows

- Setup. $\mathcal{A}$ first provides $\mathcal{F}$ with the public parameter $(e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{\mathrm{pub}})$, where $P_{\mathrm{pub}} = X$.
- H_1-queries. To respond to these queries, $\mathcal{A}$ maintains a list of tuples $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$ as explained below. We refer to this list as H_1-list. The list is initially empty. When $\mathcal{F}$ queries the oracle H_1 at an identity ID_i, $\mathcal{A}$ responds as follows.
  1. If the query ID_i already appears on the H_1-list in a tuple $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$, then $\mathcal{A}$ responds with $H_1(ID_i)$.

2. Otherwise, $\mathcal{A}$ generates a random coin $c_i \in \{0,1\}$ such that $\Pr[c_i = 0] = 1/(q_E+1)$. Then $\mathcal{A}$ randomly selects $a_i, b_i \in_R \mathbb{Z}_q^*$.
3. If $c_i = 0$, $\mathcal{A}$ sends $H_1(ID_i) = a_i Y$ to $\mathcal{F}$. Then, $\mathcal{A}$ computes $B_i = b_i P - X$, $C_i = b_i a_i Y$. Next, $\mathcal{A}$ appends the tuple $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$ to the H_1-list.
4. If $c_i = 1$, $\mathcal{A}$ sends $H_1(ID_i) = a_i P$ to $\mathcal{F}$. Then $\mathcal{A}$ computes $B_i = b_i P$, $C_i = b_i H_1(ID_i) + a_i X$. Next, $\mathcal{A}$ appends the tuple $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$.

- Extract queries. Suppose that $\mathcal{F}$ makes a extract query on the identity ID_i. $\mathcal{A}$ answers this query as follows:
  1. $\mathcal{A}$ runs the above algorithm for responding to H_1-queries to obtain $H_1(ID_i)$. Let $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$ be the corresponding tuple on the H_1-list.
  2. If $c_i = 0$, then $\mathcal{A}$ reports failure and terminates.
  3. If $c_i = 1$, $\mathcal{A}$ sends to $\mathcal{F}$ the private key $D_{ID_i} = xH_1(ID_i) = a_i X$.
- H_2-queries. To respond to these queries, $\mathcal{A}$ maintains a list of tuples in the form $(ID_i, m_i, d_i P + (1-e_i)Z, d_i, e_i)$ as explained below. We refer to this list as H_2-list. The list is initially empty. When $\mathcal{F}$ queries the oracle H_2 at an ID-message pair (ID_i, m_i), $\mathcal{A}$ responds as follows.
  1. If the query (ID_i, m_i) already appears on the H_2-list in a tuple $(ID_i, m_i, d_i P + (1-e_i)Z, d_i, e_i)$, then $\mathcal{A}$ responds with $H_2(ID_i, m_i) = d_i P + (1-e_i)Z$.
  2. Otherwise, $\mathcal{A}$ generates a random coin $e_i \in \{0,1\}$ such that $\Pr[e_i = 0] = 1/(q_s+1)$. Then $\mathcal{A}$ randomly selects $d_i \in \mathbb{Z}_q^*$, sets $H_2(ID_i, m_i) = d_i P + (1-e_i)Z$ and sends it to $\mathcal{F}$. Additionally, $\mathcal{A}$ adds the tuple $(ID_i, m_i, d_i P + (1-e_i)Z, d_i, e_i)$ to the H_2-list.
- Signature queries. Suppose that $\mathcal{F}$ chooses the identity ID_i and the plaintext m_i and wants to obtain the signature on m_i with respect to the identity ID_i. $\mathcal{A}$ answers this query as follows.
  1. $\mathcal{A}$ runs the above algorithm for responding to H_2-queries to obtain $H_2(ID_i, m_i)$, and the algorithm for responding to H_1-queries to obtain $H_1(ID_i)$. Let $(ID_i, m_i, d_i P + (1-e_i)Z, d_i, e_i)$ be the corresponding tuple on the H_2-list, and $(ID_i, H_1(ID_i), a_i, b_i, c_i, B_i, C_i)$ be the corresponding tuple on the H_1-list.
  2. If $e_i = 1$ and $c_i = 0$, $\mathcal{A}$ sends to $\mathcal{F}$ the signature $\sigma_i = (A_i, B_i, C_i)$, where $A_i = d_i(b_i P - X)$. We can easily see that the simulated signature can pass the verification, since

$$e(A_i, P) = e\big(d_i(b_i P - X), P\big) = e(d_i P, b_i P - X) = e\big(H_2(ID_i, m_i), B_i\big),$$
$$e\big(H_1(ID_i), X + B_i\big) = e\big(H_1(ID_i), b_i P\big) = e\big(b_i H_1(ID_i), P\big) = e(C_i, P).$$

  3. If $e_i = 1$ and $c_i = 1$, $\mathcal{A}$ sends to $\mathcal{F}$ the signature $\sigma_i = (A_i, B_i, C_i)$, where $A_i = d_i b_i P$. We can easily see that the simulated signature can pass the verification, since

$$e(A_i, P) = e(d_i b_i P, P) = e(d_i P, b_i P) = e\big(H_2(ID_i, m_i), B_i\big),$$

$$e\big(H_1(ID_i), X + B_i\big) = e\big(H_1(ID_i), X + b_iP\big) = e\big((x + b_i)H_1(ID_i), P\big)$$
$$= e\big(xa_iP + b_iH_1(ID_i), P\big) = e\big(a_iX + b_iH_1(ID_i), P\big) = e(C_i, P).$$

4. If $e_i = 0$, $\mathcal{A}$ reports failure and terminates.

- Outputs. Eventually algorithm $\mathcal{F}$ produces a identity-message-signature tuple (ID_f, m_f, σ_f), where $\sigma_f = (A_f, B_f, C_f)$, no signature query was issued for (ID_f, m_f) and no extract queried was issued for ID_f. To compute $e(P, P)^{xyz}$, $\mathcal{A}$ proceeds as follows.
  1. If there is no tuple on the H_2-list containing (ID_f, m_f) then $\mathcal{A}$ issues a query itself for $H_2(ID_f, m_f)$ to ensure that such a tuple exists. If there is no tuple on the H_1-list containing ID_f then $\mathcal{A}$ issues a query itself for $H_1(ID_f)$ to ensure that such a tuple exists. Let $(ID_f, m_f, d_fP + (1 - e_f)Z, d_f, e_f)$ be the corresponding tuple on the H_2-list and $(ID_f, H_1(ID_f), a_f, b_f, c_f, B_f, C_f)$ be the corresponding tuple on the H_1-list.
  2. If σ_f is not a valid signature with respect to the ID-message pair (ID_f, m_f), then $\mathcal{A}$ reports failure and terminates.
  3. If c_f = 1 or $e_f = 1$, then $\mathcal{F}$ reports failure and terminates.
  4. If c_f = 0 and $e_f = 0$, $\mathcal{F}$ outputs

$$e(P, P)^{xyz} = e(Z + d_fP, a_f^{-1}C_f)e(d_fX + A_f, -Y).$$

Below, we show the correctness of the above formulation for computing $e(P, P)^{xyz}$. Since (A_f, B_f, C_f) is a valid signature with respect to (ID_f, m_f), we have

$$e(A_f, P) = e\big(H_2(ID_f, m_f), B_f\big), \qquad e\big(H_1(ID_f), X + B_f\big) = e(C_f, P).$$

So, if we suppose $B_f = x_{ID_f}P$, then we have

$$A_f = x_{ID_f}H_2(ID_f, m_f), \qquad C_f = (x + x_{ID_f})H_1(ID_f).$$

Since $H_2(ID_f, m_f) = Z + d_fP$, $H_1(ID_f) = a_fY$, we have

$$\begin{aligned} e(Z + d_fP, C_f) &= e(Z + d_fP, (x + x_{ID_f})a_fY) \\ &= e(Z, xY)^{a_f}e(d_fP, a_fxY)e(Z + d_fP, a_fx_{ID_f}Y) \\ &= e(P, P)^{a_fxyz}e(d_fxP, Y)^{a_f}e\big(x_{ID_f}(Z + d_fP), Y\big)^{a_f} \\ &= e(P, P)^{a_fxyz}e(d_fX + A_f, Y)^{a_f}. \end{aligned}$$

Next, we have

$$\begin{aligned} e(P, P)^{xyz} &= e\big(Z + d_fP, C_f\big)^{a_f^{-1}}e(d_fX + A_f, Y\big)^{-1} \\ &= e(Z + d_fP, a_f^{-1}C_f)e(d_fX + A_f, -Y). \end{aligned}$$

This completes the description of algorithm $\mathcal{A}$. It remains to show that $\mathcal{A}$ solves the given instance of the BDH problem on $(\mathbb{G}_1, \mathbb{G}_2)$ with probability at least ϵ'. To do so, we analyze the four events needed for $\mathcal{A}$ to succeed:

- E_1: $\mathcal{A}$ does not abort as a result of any of $\mathcal{F}$'s extract queries.
- E_2: $\mathcal{A}$ does not abort as a result of any of $\mathcal{F}$'s signature queries.
- E_3: $\mathcal{F}$ generates a valid message-signature forgery (ID_f, m_f, σ_f).
- E_4: Event E_3 occurs, $c_f = 0$ for the tuple containing ID_f on the H_1-list, and $e_f = 0$ for the tuple containing (ID_f, m_f) on the H_2-list.

$\mathcal{F}$ succeeds if all of these events happen. The probability $\Pr[E_1 \wedge E_2 \wedge E_4]$ is

$$\Pr[E_1 \wedge E_2 \wedge E_4] = \Pr[E_1 \wedge E_2] \Pr\left[E_3 | E_1 \wedge E_2\right] \Pr\left[E_4 | E_1 \wedge E_2 \wedge E_3\right].$$

The following claims give a lower bound for each of these terms.

Lemma 1. *The probability that algorithm $\mathcal{A}$ does not abort as a result of $\mathcal{F}$'s extract queries or $\mathcal{F}$'s signature queries is at least $\frac{1}{e^2}$. Hence,* $\Pr[E_1 \wedge E_2] \geqslant \frac{1}{e^2}$.

Proof. Without loss of generality we assume that $\mathcal{F}$ does not ask for the signature of the same ID-message pair twice. We prove by induction that after $\mathcal{F}$ makes i signature queries, the probability that $\mathcal{A}$ does not abort is at least $(1 - \frac{1}{q_S+1})^i$. The claim is trivially true for $i = 0$. Let (ID_i, m_i) be $\mathcal{A}$'s ith signature query and let $(ID_i, m_i, d_i P + e_i Z, d_i, e_i)$ be the corresponding tuple on the H_1-list. Then prior to issuing the query, the bit e_i is independent of $\mathcal{F}$'s view – the only value that could be given to $\mathcal{F}$ that depends on e_i is $H_2(ID_i, m_i)$, but the distribution on $H_2(ID_i, m_i)$ is the same whether $e_i = 0$ or $e_i = 1$. Therefore, the probability that this query causes $\mathcal{F}$ to abort is at most $1/(q_S + 1)$. Using the inductive hypothesis and the independence of e_i , the probability that $\mathcal{A}$ does not abort after this query is at least $(1 - 1/(q_S + 1))^i$. This proves the inductive claim. Since $\mathcal{F}$ makes at most q_S signature queries the probability that $\mathcal{A}$ does not abort as a result of a signature query is at least $(1-1/(q_S+1))^{q_S} \geqslant 1/e$. Similarly, we can prove that the probability that $\mathcal{A}$ does not abort as a result of an extract query is at least $(1 - 1/(q_E + 1))^{q_E} \geqslant 1/e$. So we prove that $\Pr[E_1 \wedge E_2] \geqslant \frac{1}{e^2}$.

Lemma 2. *If algorithm $\mathcal{A}$ does not abort as a result of $\mathcal{F}$'s extract queries or signature queries, then algorithm $\mathcal{F}$'s view is identical to its view in the real attack. Hence,* $\Pr[E_3 | E_1 \wedge E_2] \geqslant \epsilon$.

Proof. The public parameters given to $\mathcal{F}$ are from the same distribution as a public key produced by algorithm Setup. Responses to H_2-queries and H_1-queries are as in the real attack since each response is uniformly and independently distributed in $\mathbb{G}_1$. All responses to signature queries and extract queries are valid. So, $\mathcal{F}$ will produce a valid message-signature pair with probability at least ϵ. Hence, $\Pr[E_3 | E_1 \wedge E_2] \geqslant \epsilon$.

Lemma 3. *The probability that algorithm $\mathcal{A}$ does not abort after $\mathcal{F}$ outputs a valid forgery is at least $\frac{1}{(q_S+1)(q_E+1)}$. Hence,* $\Pr[E_4 | E_1 \wedge E_2 \wedge E_3] = \frac{1}{(q_S+1)(q_E+1)}$.

Proof. Given that events E_1, E_2 and E_3 happened, algorithm $\mathcal{A}$ will abort only if $\mathcal{F}$ generates a forgery (ID_f, m_f, σ_f) for which the tuple $(ID_f, m_f, d_f P + e_f Z, d_f, e_f)$ on H_2-list has $e_f = 1$ or the tuple $(ID_f, H_1(ID_f), a_f, b_f, c_f, B_f, C_f)$ on the H_1-list has $c_f = 1$. At the time $\mathcal{F}$ generates its output, it knows the value of e_i for those (ID_i, m_i) for which it issued a signature query. All the remaining e_i's are independent of $\mathcal{A}$'s view. Indeed, if $\mathcal{F}$ did not issue a signature query for (ID_i, m_i) then the only value given to $\mathcal{F}$ that depends on e_i is $H_2(ID_i, m_i)$, but the distribution on $H_2(ID_i, m_i)$ is the same whether $e_i = 0$ or $e_i = 1$. Since $\mathcal{F}$ could not have issued a signature query for (ID_f, m_f) we know that e_f is independent of $\mathcal{A}$'s current view and therefore $\Pr[e_f = 0 | E_1 \wedge E_2 \wedge E_3] = 1/(1 + q_S)$ as required. Similar, we can prove that $\Pr[c_f = 0 | E_1 \wedge E_2 \wedge E_3] = 1/(1 + q_E)$. So we have $\Pr[E_4 | E_1 \wedge E_2 \wedge E_3] = \frac{1}{(1+q_S)(1+q_E)}$.

Using the bounds from the above lemmas shows that $\mathcal{A}$ produces the correct answer with probability at least $\epsilon/(\mathrm{e}^2(q_S+1)(q_E+1)) \geqslant \epsilon'$ as required. Algorithm $\mathcal{A}$'s running time is the same as $\mathcal{F}$'s running time plus the time it takes to respond to $(q_{H_1} + q_S + q_E)$ H_1-hash queries, $(q_{H_2}+q_S)$ H_2-hash queries, q_E extract queries and q_S signature queries. Each query requires at most four exponentiations in $\mathbb{G}_1$ which we assume takes time $c_{\mathbb{G}_1}$. Hence, the total running time is at most $t+4c_{\mathbb{G}_1}(q_{H_1}+q_{H_2}+3q_S+2q_E) \leqslant t'$ as required. This completes the proof of Theorem 1.

Theorem 2. *The proposed threshold ID-based signature scheme is simulatable.*

Proof. First, to prove the simulability of the private key distributing algorithm ShareKey, we construct the simulator SIM_1 as follows. Without loss of generality, assume that the parties indexed $1, 2, \ldots, t$ have been corrupted by the simulator SIM_1. SIM_1 is given the public parameter $params = (\mathbb{G}_1, \mathbb{G}_2, e, q, P, P_{\mathrm{pub}}, H_1, H_2)$ and the identity ID.

- SIM_1 selects a random $c \in \mathbb{Z}_q^*$ and sets $C = cH_1(ID)$, $B = cP - P_{\mathrm{pub}}$. Then, SIM_1 broadcasts B, C.
- SIM_1 selects t random number $x_{ID,1}, \ldots, x_{ID,t} \in \mathbb{Z}_q^*$, and sets $B_0 = B, B_1 = x_{ID,1}P, \ldots, B_t = x_{ID,t}P$. For $B_0, B_1, \ldots, B_t$, there is only one t-degree polynomial $f(x) = a_0 + a_1 x + \cdots + a_t x^t$ such that $B_0 = f(0)P, B_1 = f(1)P, \ldots, B_t = f(t)P$. SIM_1 computes and broadcasts such $a_0P, a_1P, \ldots, a_tP$. Then SIM_1 sends $x_{ID,i}$ to the corrupted party indexed by i, for $i = 1, \ldots, t$. Below, we show the computability of the values $a_0P, a_1P, \ldots, a_tP$. We have $(B_0, B_1, \ldots, B_t) = (a_0P, a_1P, \ldots, a_tP)T$, where T is an invertible matrix as follows:

$$\begin{pmatrix} 1 & 1^0 & 2^0 & \ldots & t^0 \\ 0 & 1^1 & 2^1 & \ldots & t^1 \\ 0 & 1^2 & 2^2 & \ldots & t^2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 1^t & 2^t & \ldots & t^t \end{pmatrix}.$$

Hence, SIM_1 can compute

$$(a_0, a_1P, \ldots, a_tP) = (B_0, B_1, \ldots, B_t)T^{-1}.$$

Second, to prove the simulability of the threshold signing protocol, we construct the simulator SIM_2 as follows. Suppose that SIM_2 is given a common parameter $params = (\mathbb{G}_1, \mathbb{G}_2, e, q, P, P_{\text{pub}}, H_1, H_2)$, an identity ID, a message m, and a signature $\sigma = (A, B, C)$ on m, t shares $x_{ID,1}, \ldots, x_{ID,t}$ of the corrupted signature generation servers, and the public outputs of KeyShare. SIM_2 computes the signature shares $A_1 = x_{ID,1}H_2(ID, m), \ldots, A_t = x_{ID,t}H_2(ID, m)$, then sets $A_i = \sum_{j=0}^{t} L_{\Phi,j}(i)A_j$, for $i = t+1, \ldots, n$, where $A_0 = A$ and $L_{\Phi,j}(i)$ is the corresponding Lagrange coefficients. Then SIM_1 plays the role of P_i to broadcasts A_i, for $i = t+1, \ldots, n$.

It is obvious that the simulated view of the t corrupted parties is completely same to that for the real running of KeyShare, TSign.

As we mentioned before, Baek and Zheng (2004) proved that if the underlying ID-based signature scheme is unforgeable and the threshold signature is simulatable, then the ID-based threshold signature scheme is unforgeable. Hence, from Theorems 1, 2, the following theorem is immediately derived.

Theorem 3. *The proposed ID-based threshold signature scheme is UF-IDTS-CMA secure in the random oracle model, under the assumption the bilinear Diffie–Hellman problem is intractable.*

Since the validity of each signature share can be publicly verified, and hence at least $t + 1$ valid signature shares can be obtained, we easily have the following theorem.

Theorem 4. *The proposed ID-based threshold signature scheme is* (t, n)*-robust, for any* t *such that* $n \geqslant 2t + 1$.

6. Efficiency Analysis

In this section, we roughly compare the efficiency of our proposed ID-based threshold signature with the other two exiting ID-based threshold signature schemes: Beak–Zheng's (2004) scheme (denoted by the BZ scheme) Baek and Zheng (2004) and Chen *et al.*'s (2004) scheme (denoted by the CZKK Scheme) which are both provably secure in the random oracle model.

First, we simply compare the BZ ID-based threshold signature scheme and the CZKK ID-based threshold signature scheme. In fact, the purpose of the CZKK scheme is to solve the key escrow problem of the ID-based threshold signature scheme. In other words, if this additional property is not taken into account, the CZKK scheme and the BZ scheme are very similar. One of the main differences between the BZ scheme and the CZKK scheme is in the sub-protocol of collective generation of the random element: the BZ scheme uses the timing-consuming protocol called distributed key generation protocol based on the bilinear map, while in the CZKK, the random element is simply the sum of all the random elements of all signing players. This difference brings the difference of the robustness property: in the CZKK scheme, the threshold signing protocol will not output a valid signature until all signature shares are valid (the validity of the signature share can

be publicly verified). However, the robustness of the BZ scheme is "true": if $n \geqslant 2t + 1$, the threshold signing protocol always outputs a valid signature when the number of the invalid signature shares is not more than t. So we can say that the CZKK scheme achieves better efficiency at the cost of loss robustness.

Now, we roughly compare the efficiency of our scheme with the BZ threshold signature scheme. In the our scheme, every signing party just broadcasts a signature share, then finds $t + 1$ valid signature shares and combines these $t + 1$ shares into a valid signature. However, in the BZ scheme, before computing the signature share, all the parties need to collectively generate a random element using the sub-protocol (Distributed Key Generation Protocol Based on the Bilinear Map (DKPBM; Baek and Zheng, 2004) in which every party runs a sub-protocol (Unconditionally Secure Verifiable Secret-Sharing Scheme Based on the Bilinear Map; Baek and Zheng, 2004) to share a random element chosen by himself. By avoiding the expensive sub-protocol of DKPBM, our ID-based threshold signature scheme achieves the following advantages:

- *Optimal Round Complexity*. The threshold signing protocol of our schemes is one round (and hence the round complexity is optimal), while the threshold signing protocol of the BZ scheme is multi-round.
- *Reduced Computation Complexity*. Of course, the sub-protocol for collectively generating the random element is the major time-consuming component in the BZ scheme. So the efficiency of our ID-based threshold signature scheme is much better than that of the BZ scheme.
- *Optimal Communication Model*. In the DKPBM, a private channel between each other is required. However, in our scheme, what every signing party does is just to broadcast his signature share and collects more than t valid signature shares.So the private channels among all players are not needed any more. In the private key distributing procedure KeyShare, however, a private channel between the dealer and each party is needed.

Also note that the robustness of our scheme is same to that of the BZ scheme.

7. Conclusion

In this paper, based on the bilinear Diffie–Hellman assumption, we proposed a new ID-based robust threshold signature scheme which is provably secure (unforgeable and robust) in the random oracle model in the presence of t malicious players if the total number $n \geqslant 2t + 1$. We shown that its efficiency is much better than the Baek–Zheng ID-based threshold signature scheme which is also provably secure in random oracle model.

References

Baek, J., Zheng, Y. (2004). Identity-based threshold signature scheme from the bilinear pairings. In: *Proc. Int. Conf. on Information Technology*: *Coding and Computing*. IEEE Computer Society Press, pp. 124–128.

Bellare, M., Namprempre, C., Neven, G. (2004). Security proofs for identity-based identification and signature schemes. In: *EUROCRYPT 2004, Lecture Notes in Computer Science*, Vol. 3027. Springer-Verlag, Berlin, pp. 268–286.

Bellare, M., Rogaway, P. (1993). Random oracles are practical: a paradigm for designing efficient protocols. In: *Proc. the 1st Annual Conf. on Computer and Communications Security*. ACM Press, New York, pp. 62–73.

Boldyreva, A. (2003). Efficient threshold signatures, multisignatures and blind signatures based on the gap-Diffie–Hellman-group signature scheme. In: *PKC 2003*, *Lecture Notes in Computer Science*, Vol. 2567. Springer-Verlag, Berlin, pp. 31–46.

Boneh, D., Franklin, M. (2001). Identity-based encryption from the Weil pairing. In: *CYPTO 2001*, *Lecture Notes in Computer Science*, Vol. 2139. Springer-Verlag, Berlin, pp. 213–229.

Cha, J.C., Cheon, J.H. (2003). An identity-based signature from gap Diffie–Hellman groups. In: *PKC 2003*, *Lecture Notes in Computer Science*, Vol. 2567. Springer-Verlag, Berlin, pp. 18–30.

Chen, X., Zhang, F., Kim, K. (2004). New ID-based threshold signature scheme from bilinear pairings. In: *INDOCRYPT 2004*, *Lecture Notes in Computer Science*, Vol. 3348. Springer-Verlag, Berlin, pp. 371–383.

Desmedt, Y. (1994). Threshold cryptography. *European Transactions on Telecommunications*, **5**(4), 449–457.

Feldman, P. (1987). A practical scheme for non-interactive verifiable secret sharing. In: *Proc. FOCS'87*. ACM Press, New York, pp. 427–437.

Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T. (2001). Robust threshold DSS signatures. *Information and Computation*, **164**(1), 54–84.

Hess, F. (2002). Efficient Identity based signature schemes based on pairings. In: *SAC 2002*, *Lecture Notes in Computer Science*, Vol. 2595. Springer-Verlag, Berlin, pp. 310–324.

Herzberg, A., Jarecki, S., Krawczyk, H., Yung, M. (1995). Proactive secret sharing, or: how to cope with perpetual leakage. In: *CRYPTO 95*, *Lecture Notes in Computer Science*, Vol. 963. Springer-Verlag, Berlin, pp. 339–352.

Joux, A. (2002). The Weil and Tate pairings as building blocks for public key cryptosystems. In: *Algorithm Number Theory Symposium – ANTS 2002*, *Lecture Notes in Computer Science*, Vol. 2369. Springer-Verlag, Berlin, pp. 20–32.

Kancharl, P.K., Gummadidala, S., Saxena, A. (2005). Identity based strong designated verifier signature scheme. *Informatica*, **16**(2), 261–274.

Qian, H., Cao, Z., Xue, Q. (2005). Efficient pairing-based threshold proxy signature scheme with known signers. *Informatica*, **16**(2), 261–274.

Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In: *Crypto 84*, *Lecture Notes in Computer Science*, Vol. 196. Springer-Verlag, Berlin, pp. 47–53.

W. Gao obtained the PhD degree in applied mathematics from School of Mathematics at Hunan University in 2006. Now he is an assistant professor at Ludong University. His current research interests focus on the areas of cryptology, electronic commerce security, and network security etc.

G. Wang obtained the PhD degree in computer application technology from Chinese Academy of Sciences in 2001. Now he is a lecturer at University of Birmingham. His current research areas include applied cryptography, information security, and electronic commerce.

X. Wang is a professor at South China Normal University and his main research interests are number theory and basic cryptography.

Z. Yang is a professor at Ludong University and his main research interests are applied algebra and cryptography.

Vieno ciklo identifikatoriumi pagrįsta slenkstinio parašo schema naudojanti bitiesinį poravimą

Wei GAO, Guilin WANG, Xueli WANG, Zhenguang YANG

Pasiūlyta nauja identifikatoriumi pagrįsta slenkstinio parašo schema, naudojanti bitiesinio poravimo metodą, kuri yra saugi, kai tenkinamos Diffie–Hellman bitiesinės sąlygos. Šioje schemoje privatusis raktas yra susietas su vartotojo tapatybe ir, palyginus su Baek ir Zheng metodu, turi tokius privalumus: (1) slenkstinio parašo protokolo sudėtingumas yra optimalus, nes pasirašymo procedūros metu kiekvienas dalyvis siunčia tik vieną pranešimą, (2) perdavimo kanalas yra optimalus, nes nereikalingas papildomas privatusis kanalas, (3) ši schema yra efektyvesnė skaičiavimų sudėtingumo prasme, nes privatusis raktas yra išplatinamas tam tikro skaičiaus pagalba, todėl nereikia vykdyti sudėtingo skaičiavimo prasme raktų generavimo protokolo.