A non-delegatable identity-based strong designated verifier signature scheme

Bin Wang

Information Engineering College of Yangzhou University

Yangzhou City, Jiangsu Province, 225009, P.R.China

E-mail:jxbin76@yahoo.cn

Abstract: In a strong designated verifier signature scheme, no third party can verify the validity of a signature. On the other hand, non-delegatability, proposed by Lipmaa, Wang and Bao, is another stronger notion for designated verifier signature schemes. In this paper, we formalize a security model for non-delegatable identity based strong designated verifier signature (IDSDVS) schemes. Then a novel non-delegatable IDSDVS scheme based on pairing is presented. The presented scheme is proved to be non-delegatable, non-transferable and unforgeable under the Gap Bilinear Diffie-Hellman assumption.

Keywords: Strong designated verifier signature, Non-delegatability, Bilinear pairing, Gap Bilinear Diffie-Hellman assumption, Random oracle model;

1. Introduction

Ordinary digital signature schemes allow a signer with a secret key to sign messages such that anyone can verify the authenticity of the signed messages via the corresponding public key. However, the public verifiability of ordinary digital signatures is not desirable in some applications when a verifier should not convince any third party about the fact by presenting a signature on a message, such as certificates for personal health records, income summary. To address this issue, in 1996, Jacobsson et al. introduced the concept of designated verifier signature(DVS) [2]. A DVS scheme makes it possible to prevent a designated verifier from transferring his conviction about validity of signed messages to any third party. The reason is that the designated verifier is able to simulate a signature that is indistinguishable from a real signature intended to him. That is, designated verifier signatures do not provide non-repudiation property of ordinary digital signatures. Designated verifier signatures have several applications, such as E-voting, call for tenders and software licensing.

Jacobsson et al. [2] also introduced a stronger notion of designated verifier signature, called strong designated verifier signature (SDVS). In a SDVS scheme, no third party can verify the validity of a signature without the knowledge of the designated verifier’s secret key. In 2003, Saeednia formalized the notion of strong designated verifier signature(SDVS) [7] and proposed an efficient scheme in their paper. Later, Susilo et al. [8] proposed an identity based SDVS scheme which is only an identity based variant of the scheme of [7].

On the other hand, Lipmaa et al. [5] described another stronger notion of designated verifier signature, called non-delegatability. That is, there exists an efficient knowledge extractor that can extract either the signer’s secret key or the designated verifier’s secret key, when given oracle access to an adversary who can create valid signatures with a high probability. Recently, Zhang and Mao [9] proposed an identity based strong designated verifier signature (IDSDVS) scheme which is claimed to offer non-delegatability. However, Kang et al. [3] showed that Zhang-Mao scheme can not satisfy the strongness property. That is, Zhang-Mao scheme allows anyone who intercepts one signature to verify subsequent signatures. In addition, Kang et al. [4] also proposed an IDSDVS scheme and claimed that the security of their scheme is related to the bilinear Diffie-Hellman problem.

Nevertheless, we show that Kang et al.’s scheme presented in [4] is also vulnerable to the attack described in [3]. The essence of this attack is that the schemes in [4, 9] are delegatable. An effective solution to this problem is to design a non-delegatable IDSDVS scheme. Although Kang et al. [3] presented an efficient IDSDVS scheme, which uses hash operations to destroy algebraic structure of the produced signature in order to avoid the attack described in [3], it is not difficult to show that their scheme is also delegatable. Furthermore, no formal security proof is presented for their scheme in [3].

Motivated by the above discussion, we formalize a security model for non-delegatable IDSDVS schemes in this paper. Then we point out that the scheme proposed in [4] is also vulnerable to the attack described in [3]. In the following, we present a non-delegatable IDSDV signature scheme based on bilinear pairing, which is secure under our security model. The security of our scheme is based on the hardness of the Gap Bilinear Diffie-Hellman problem. However，the security result of our scheme is not tight as the reduction uses the Forking Lemma. It remains to be seen if there exists a non-delegatable IDSDVS scheme that enjoys a tight reduction without using the Forking Lemma. Finally, we compare the efficiency with other related IDSDVS schemes.

2. Preliminaries

2.1 Bilinear pairing

Let $< G_1, + >$ be a cyclic additive group generated by P, whose order is a large prime q, $< G_2, \bullet >$ be a cyclic multiplicative group of the same order, and let $e : G_1 \times G_1 \to G_2$ be a bilinear pairing with the following properties:

1. Bilinear: For any Q , R , T $\in G_1$, $e(Q+R,T) = e(Q,T) \bullet e(R,T)$ and $e(Q,R+T) = e(Q,R) \bullet e(Q,T)$

2. Non-degenerate: There exists $R, T \in G_1$, such that $e(R,T) \neq 1$

3. Computable: There exists an efficient algorithm to compute $e(R,T)$ for any $R, T \in G_1$.

2.2 Notation

If A is a randomized algorithm, then $y \leftarrow A^{O_1(\cdot), O_2(\cdot), \cdots}(x_1, x_2, \cdots)$ means that A has input $x_1, x_2, \cdots$, access to oracles $O_1, O_2, \cdots$, and the output of A is assigned to y. We use the notation $x \in_R S$ to mean "the element x is chosen with uniform probability from the set S".

2.3 Complexity assumptions

Definition 1: Bilinear Diffie-Hellman(BDH) Problem in (G_1, G_2) **:** Given $P, a \cdot P, b \cdot P, c \cdot P \in G_1$ for some unknown $a, b, c \in Z_q$, compute $d = e(P,P)^{abc} \in G_2$.

Definition 2: Decisional Bilinear Diffie-Hellman(DBDH) Problem in (G_1, G_2) **:** Given $P, a \cdot P, b \cdot P, c \cdot P \in G_1, z \in G_2$ for some unknown $a, b, c \in Z_q$, decide whether

$z = e(P,P)^{abc}$ holds.

Definition 3: Gap Bilinear Diffie-Hellman(GBDH) Problem in (G_1, G_2) **:** Given $P, a\cdot P, b\cdot P, c\cdot P \in G_1$ for some unknown $a,b,c \in Z_q$, compute $d = e(P,P)^{abc} \in G_2$ with the help of a DBDH oracle O_{DBDH}.

Remark: A DBDH oracle O_{DBDH} outputs 1 if $z = e(P,P)^{abc}$ and 0 otherwise.

The success probability of an algorithm A in solving the GBDH problem in (G_1, G_2) is $Succ_{G_1,G_2}^{GBDH}(A) = \Pr[A^{O_{DBDH}}(P, a\cdot P, b\cdot P, c\cdot P) = e(P,P)^{abc} : a,b,c \in Z_q]$.

A (t,ε)-GBDH solver A is a probabilistic polynomial-time algorithm running in time at most t such that the success probability $Succ_{G_1,G_2}^{GBDH}(A) \geq \varepsilon$. We say that (G_1, G_2) satisfies the GBDH assumption if there is no polynomial time (t,ε)-GBDH solver A with advantage ε non-negligible.

3. Weakness of Kang et al's scheme [4]

3.1 Review of Kang et al's scheme [4]

1. Setup: Let $< G_1, + >$ be a gap Diffie-Hellman group generated by P, whose order is a large prime q, $< G_2, \bullet >$ be a cyclic multiplicative group of the same order, and let $e: G_1 \times G_1 \to G_2$ be a bilinear map. Then PKG (private key generation centre) picks a random number $s \in Z_q^*$ as the master secret key and sets the master public key to $P_{pub} = s \cdot P$. H_1, H_2 are two cryptographic hash functions

$$H_1 : \{0,1\}^* \to G_1,\ H_2 : \{0,1\}^* \to Z_q^*.$$

The public system parameters are $< G_1, G_2, P, P_{pub}, H_1, H_2, e, q >$.

2.KeyExtract: Given an identity ID_i, PKG computes $Q_i = H_1(ID_i)$, $S_i = s \cdot Q_i$. Then KGC distributes the secret key S_i to the corresponding user identified by ID_i over a secure channel.

3.Sign: To sign a message m intended for a verifier Bob with identity ID_B, a signer Alice with identity ID_A picks a random number $k \in Z_q^*$ and computes

$$t = e(P, Q_B)^k, \quad T = k \cdot P + H_2(m,t)S_A, \quad \sigma = e(T, Q_B).$$

The signature on m is (σ, t).

4.Verify: Given the system parameters, $Q_A = H_1(ID_A)$, and the signature (σ, t) on the signed message m, the correctness of (σ, t) can be verified by Bob as follows:

$$\sigma = t \cdot e(Q_A, S_B)^{H_2(m,t)}$$

5.Signature Simulation: Bob can produce the simulated signature $\widehat{\sigma}$ intended for him as follows:

(1) Picks a random number $\hat{k} \in Z_q^*$.

(2) Computes $\hat{t} = e(P, Q_B)^{\hat{k}}$, $\widehat{\sigma} = \hat{t} \cdot e(Q_A, S_B)^{H_2(m,\hat{t})}$.

3.2 Attack on Kang et al’s scheme [4]

An adversary who intercepts the signature (σ, t) can compute $e(Q_A, S_B)$ as follows:

$$e(Q_A, S_B) = (\sigma/t)^{(H_2(m,t))^{-1}}.$$

Thus it is easy to see that the adversary can verify the correctness of subsequent signatures and simulate valid signatures intended for Bob via $e(Q_A, S_B)$. The essence of this attack is that Kang et al’s scheme [4] is delegatable. That is, anyone who has the knowledge of the trapdoor $e(Q_A, S_B)$ can verify the correctness of signatures and simulate valid signatures. An effective solution to this problem is to design a non-delegatable IDSDVS scheme. The notion of delegatability is discussed in subsection 4.2.3.

Although Kang et al presented another efficient scheme in [3] and claimed it to be secure against this attack, it is not difficult to show that their scheme is delegatable. So we omit the details to show that their scheme in [3] is delegatable. In addition, no formal security proof is presented for their scheme in [3]. In the following, we focus on establishing a security model for non-delegatable IDSDVS schemes and present a scheme secure under this model.

4. Formal model of IDSDVS schemes

4.1 Definition of IDSDVS schemes

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

1. MasterKeyGen(Master Key Generation**):** On input a security parameter $k \in \mathrm{N}$, it generates a list of system parameters **params,** and a master public/secret key pair (mpk, msk). This algorithm is assumed to be run by a Key Generation Center (KGC).

2. KeyGen(User Key Generation**):** On input msk, an identity $ID \in \{0,1\}^*$, it generates a secret key $sk_{ID} \leftarrow$ **KeyGen** (ID, msk). This algorithm is run by the KGC for each user and the generated secret key is assumed to be distributed securely to the corresponding user.

3. SDV_Sign(Signature Generation**):** On input a signer's secret key sk_{ID_S}, a verifier's identity ID_D and a message m, it generates a signature by executing $\sigma_{DV} \leftarrow$ S**DV_Sign** (sk_{ID_S}, ID_D, m). We require that the signer's identity $ID_S \neq ID_D$ since it is meaningless to generate a signature to be verified only by the signer.

4. SDV_Verf(Signature Verification**):** On input the signer's identity ID_S, the verifier's secret key sk_{ID_D}, the signed message m and the signature σ_{DV}, **SDV_Verf** returns 1 if σ_{DV} is accepted, and 0 otherwise.

5. SDV_Sim(Signature Simulation**):** On input the signer's identity ID_S, the verifier's secret key sk_{ID_D}, and a message m, it generates a simulated signature $\overline{\sigma_{DV}} \leftarrow$ **SDV_Sim** (sk_{ID_D}, ID_S, m).

Consistency: $\forall\ m \in \{0,1\}^*$, $ID \in \{0,1\}^*$, $k \in \mathrm{N}$, $(mpk, msk) \leftarrow$ **MasterKeyGen** (1^k), $sk_{ID} \leftarrow$ **KeyGen** (ID, msk), the following hold:

(1) $\forall\ \sigma_{DV} \leftarrow$ **SDV_Sign** (sk_{ID_S}, ID_D, m), Pr[SDV_Verf $(sk_{ID_D}, ID_S, m, \sigma_{DV})$=1]=1.

(2) $\forall\ \overline{\sigma_{DV}} \leftarrow$ **SDV_Sim** (sk_{ID_D}, ID_S, m), Pr[SDV_Verf $(sk_{ID_D}, ID_S, m, \overline{\sigma_{DV}})$=1]=1.

4.2 Security model

4.2.1 Unforgeability

Let IDSDVS be an identity-based strong designated verifier signature scheme and $k \in \mathrm{N}$ be a security parameter. In this section, we define the existential unforgeability of IDSDVS schemes against an adaptive chosen message and chosen identity adversary A as follows. Define a game $Exp^{CMA,CID}_{EUF,IDSDVS}(A,k)$ in which the adversary A interacts with a game challenger S.

Phase 1: At first, S runs **MasterKeyGen** (1^k) to get (mpk, msk) and a list of system parameters **params**. In the following, S picks n identities $X = \{ID_1, \cdots, ID_n\}$. Then $Corr$ is initialized to an empty set $\varnothing$ which is used to keep track of those corrupted users' identities. S gives mpk, **params** and X to the adversary A.

Phase 2: A issues the following queries:

1. UserKey queries: On input an identity $ID \in X$ chosen by A, S runs **KeyGen** (ID, msk) to get the secret key sk_{ID} and returns sk_{ID} to A. Then $Corr \leftarrow Corr \bigcup \{ID\}$.

2. Sign queries: On input a signer's identity ID_i , a verifier's identity ID_j and a message m adaptively chosen by A, S returns a signature by executing $\sigma_{DV} \leftarrow$ S**DV_Sign** (sk_{ID_i}, ID_j, m).

3. Verf queries: On input the signer's identity ID_i , the verifier's identity ID_j, and a message/signature pair (m, σ_{DV}) provided by A, S returns the result of running SDV_Verf $(sk_{ID_j}, ID_i, m, \sigma_{DV})$.

4. Sim queries: On input the signer's identity ID_i , the verifier's identity ID_j and a message m adaptively chosen by A, S returns the simulated signature

$\overline{\sigma_{DV}} \leftarrow \textbf{SDV\_Sim}(sk_{ID_j}, ID_i, m)$.

Phase 3: A wins the game if A outputs ID_S (the signer's identity), ID_D (the verifier's identity) and a message/signature pair (m^*, σ^*_{DV}) such that:

(1) $ID_S \neq ID_D$, SDV_Verf $(sk_{ID_D}, ID_S, m^*, \sigma^*_{DV})$=1.

(2) A never made a **Sign** query or a **Sim** query on (ID_S, ID_D, m^*) and $Corr \bigcap \{ID_S, ID_D\} = \varnothing$.

We define the success probability of the adversary as

$$Succ^{CMA,CID}_{EUF,IDSDVS}(A,k) = \Pr[Exp^{CMA,CID}_{EUF,IDSDVS}(A,k) = 1].$$

An IDSDVS scheme is existential unforgeable against chosen message and chosen identity attack if the success probability $Succ^{CMA,CID}_{EUF,IDSDVS}(A,k)$ is negligible for any probabilistic polynomial time (**PPT**) adversary A.

Remark: Zhang-Mao scheme [9] only provided an informal definition of unforgeability. An adversary in the security model of [4] is not allowed to issue Sim queries when attacking unforgeability. In contrast, our model provides a stronger security notion in which an adversary is allowed to issue Sim queries when attacking unforgeability.

4.2.2 Non-transferability

Non-transferability means that a third party is not able to determine whether a message is signed by the signer, or is simulated by the designated verifier. Formally, let IDSDVS be an identity-based strong designated verifier signature scheme, and $k \in \mathrm{N}$ be a security parameter. The non-transferability of IDSDVS schemes against adaptive chosen message and chosen identity distinguisher D can be defined as follows. Define a game $Exp^{CMA,CID}_{Non-Tran,IDSDVS}(D,k)$ in which the distinguisher D interacts with a game challenger S.

Phase 1: At first, S runs **MasterKeyGen**(1^k) to get (mpk, msk) and a list of system parameters **params**. In the following, S picks n identities $X = \{ID_1, \cdots, ID_n\}$. Then $Corr$ is initialized to an empty set $\varnothing$ which is used to keep track of the corrupted users' identities. S gives mpk, **params** and X to D.

Phase 2: UserKey queries, Sign queries, Verf queries and **Sim queries** issued by D are the same as those defined in section 4.2.1.

Challenge: Once D decides that Phase 2 is over, D picks a tuple (ID_S^*, ID_D^*, m^*) such that (ID_S^*, ID_D^*, m^*) has not been submitted as one of the **Sign** queries, **Sim** queries. Moreover, it is required that $Corr \bigcap \{ID_S^*, ID_D^*\} = \varnothing$. Then the challenger S picks a random bit $b \in \{0,1\}$. If $b = 0$, S returns a real signature $\sigma_{DV} \leftarrow$ S**DV_Sign** $(sk_{ID_S^*}, ID_D^*, m^*)$ to the distinguisher D. Otherwise, S returns a simulated signature $\overline{\sigma_{DV}} \leftarrow$ **SDV_Sim** $(sk_{ID_D^*}, ID_S^*, m^*)$ to the distinguisher D.

Phase 3: Upon receiving the challenging response from S, D still makes **UserKey** queries**, Sign** queries, **Sim** queries and **Verf** queries except that he cannot submit (ID_S^*, ID_D^*, m^*) as one of the **Sign** queries, **Sim** queries.

Guess: Finally, D outputs a bit $b^/$. D wins the game if $b^/ = b$.

The advantage of D in this game is $Adv(D,k) = |\Pr[b^/ = b] - \frac{1}{2}|$. An IDSDVS scheme is non-transferable against adaptive chosen message and chosen identity distinguisher D if for any probabilistic polynomial time (**PPT**) D, the advantage $Adv(D,k)$ is negligible.

4.2.3 Non-delegatability

The definition of non-delegatability is presented in [5]. We provide a straightforward adaptation of [5] to IDSVDS schemes. A delegatable IDSVDS scheme means that the signer identified by ID_S, without disclosing his secret key sk_S, may disclose some side information $y_{S,D} = f(sk_S, ID_D)$ to an adversary such that the adversary can produce valid signatures on behalf of ID_S such that these signatures can be verified only by the verifier identified by ID_D. Similarly, the verifier identified by ID_D may disclose some side information $y_{D,S} = f(sk_D, ID_S)$ to the adversary such that the adversary can produce valid simulated signatures.

In the original definition of designated verifier proofs [2], a proof of the truth of some statement Φ is a designated verifier proof if it is a proof that either Φ is true or the prover knows the secret key of the verifier. Clearly, this requirement is not satisfied by delegatable IDSVDS schemes since a signer only proves that either Φ is true or he knows some side information $y_{S,D}$ or $y_{D,S}$. We formalize the definition of non-delegatable IDSVDS schemes via an alternative formulation from [1] as follows:

Let $k \in [0,1]$ be the knowledge error and (sk_S, ID_S) (resp., (sk_D, ID_D)) be the secret/public key pair of the signer(resp., verifier). Assume that there is an algorithm F that can produce a valid signature σ on input a message m such that SDV_Verf $(sk_{ID_D}, ID_S, m, \sigma)=1$ with probability $\varepsilon > k$. We say that an IDSVDS scheme is (τ, k) non-delegatable if there is a knowledge extractor K that runs in expected polynomial time (without counting the time to make the oracle queries) with access to the oracle $F(\bullet)$ such that:

$$\Pr[x \in \{sk_{ID_S}, sk_{ID_D}\} : x \leftarrow K^{F(\cdot)}(\bullet)] \geq (\varepsilon - k)/\tau .$$

5. Our scheme

In this section, we propose a non-delegatable IDSDVS scheme as follows:

1. MasterKeyGen(Master Key Generation**):** On input a security parameter $k \in \mathrm{N}$, let $< G_1, + >$ be a cyclic additive group generated by P, whose order is a large prime q, $\log_2 q \approx k$, and $< G_2, \bullet >$ be a cyclic multiplicative group of the same order. $e : G_1 \times G_1 \to G_2$ represents a bilinear map. Then KGC performs the following operations:

(1) Picks a random number $s \in Z_q^*$ and sets the master public/secret key pair $< mpk, msk > = < s \cdot P, s >$.

(2) Chooses two collision-resistant hash functions $G:\{0,1\}^* \rightarrow Z_q, H:\{0,1\}^* \rightarrow G_1$.

(3) Sets the system parameters **params** to $<(G_1,+),(G_2,\bullet),e,q,P,G,H>$.

KeyGen(User Key Generation)**:** On input an identity ID_i, KGC computes $Q_i = H(ID_i)$, $sk_i = msk \cdot Q_i$. Then KGC distributes sk_i to the corresponding user as his secret key over a secure channel. The user can verify the correctness by checking $e(sk_i, P) = e(Q_i, mpk)$.

SDV_Sign(Signature Generation**):** Given the signer's key pair (sk_{ID_S}, ID_S), the verifier's identity ID_D and a message m, the signer should perform the following steps:

(1) Picks random numbers $r_S, w_S, t_S \in Z_q$ and computes

$$R_S = e(Q_{ID_D}, r_s \cdot P),\ T_S = t_S \cdot mpk.$$

(2) Computes $V_S = e(mpk, t_S \cdot P + w_S \cdot Q_{ID_D})$, $h_S = G(ID_S, ID_D, R_S, V_S, m)$.

(3) Computes $Z_S = r_S \cdot P + (h_S + w_S) \cdot sk_{ID_S}$

(4) The signature is $\sigma_{DV} = < R_S, w_S, T_S, Z_S >$.

SDV_Verf(Signature Verification**):** Given the signer's identity ID_S, the verifier's key pair (sk_{ID_D}, ID_D), the signed message m and the corresponding signature σ_{DV}, the correctness of $\sigma_{DV} = < R_S, w_S, T_S, Z_S >$ can be verified as follows:

(1) Computes $V_S = e(P, T_S + w_S \cdot sk_{ID_D})$, $h_S = G(ID_S, ID_D, R_S, V_S, m)$

(2) Returns 1 if and only if $e(Q_{ID_D}, Z_S) = R_S \bullet e(sk_{ID_D}, (h_S + w_S) \cdot Q_{ID_S})$.

It is easy to check the correctness of the above verification process as follows:

$$e(P, T_S + w_S \cdot sk_{ID_D}) = e(P, s \cdot (t_S \cdot P + w_S \cdot Q_{ID_D}))$$

$$= e(mpk, t_S \cdot P + w_S \cdot Q_{ID_D}) = V_S$$

$$e(Q_{ID_D}, Z_S) = e(Q_{ID_D}, r_S \cdot P + (h_S + w_S) \cdot sk_{ID_S})$$

$$= e(Q_{ID_D}, r_S \cdot P)\ e(Q_{ID_D}, (h_S + w_S) \cdot sk_{ID_S})$$

$$= R_S \bullet e(sk_{ID_D}, (h_S + w_S) \cdot Q_{ID_S})$$

SDV_Sim(Signature Simulation**):** Given the signer's identity ID_S, the verifier's key pair (sk_{ID_D}, ID_D), a message m, a simulated signature $\overline{\sigma_{DV}}$ can be generated as follows:

(1) Picks random α_D , $\lambda_D \in Z_q$, $Z_D \in G_1$ and computes

$V_D = e(P, \alpha_D \cdot P)$.

(2) Computes $R_D = e(Q_{ID_D}, Z_D) \bullet e(sk_{ID_D}, -\lambda_D \cdot Q_{ID_S})$, $h_D = G(ID_S, ID_D, R_D, V_D, m)$.

(3) Computes $w_D = (\lambda_D - h_D) \bmod q$.

(4) Computes $T_D = (\alpha_D \cdot P - w_D \cdot sk_{ID_D})$.

(5) The simulated signature is $\overline{\sigma_{DV}} = < R_D, w_D, T_D, Z_D >$

The correctness of $\overline{\sigma_{DV}}$ can be checked as follows:

$e(P, T_D + w_D \cdot sk_{ID_D}) = e(P, \alpha_D \cdot P) = V_D$

$e(Q_{ID_D}, Z_D) = R_D \bullet e(sk_{ID_D}, \lambda_D \cdot Q_{ID_S}) = R_D \bullet e(sk_{ID_D}, (h_D + w_D) \cdot Q_{ID_S})$

6.Security Analysis

Lemma 1: Given the key pairs (sk_{ID_S}, ID_S), (sk_{ID_D}, ID_D), the following distributions are indistinguishable for a polynomial-time adversary in the random oracle model.

$$\delta = \left\{ (R_S, w_S, T_S, Z_S) \middle| \begin{array}{l} R_S \in_R G_2, w_S \in_R Z_q \\ T_S \in_R G_1, h_S \in_R Z_q \\ Z_S = r_S \cdot P + (h_S + w_S) \cdot sk_{ID_S} \end{array} \right\}$$

$$\delta' = \left\{ (R_D, w_D, T_D, Z_D) \middle| \begin{array}{l} Z_D \in_R G_1, \alpha_D \in_R Z_q, R_D \in_R G_2 \\ h_D \in_R Z_q, w_D \in_R Z_q \\ T_D = (\alpha_D \cdot P - w_D \cdot sk_{ID_D}) \end{array} \right\}$$

Proof: At first, we choose a valid tuple $\sigma' = (R, w, T, Z)$ such that for some message m SDV_Verf $(sk_{ID_D}, ID_S, m, \sigma')$=1. In other words, the following equations hold:

$$V = e(P, T + w \cdot sk_{ID_D})\ , h = G(ID_S, ID_D, R, V, m)$$

$$e(Q_{ID_D}, Z) = R \bullet e(sk_{ID_D}, (h+w) \cdot Q_{ID_S})\ .$$

We then compute the probability of appearance of this tuple following each distribution of probabilities. For the sake of simplicity, we will omit the notation $\bmod q$ in the rest of the proof.

Claim 1: $\Pr_{\delta}[(R_S, w_S, T_S, Z_S) = (R, w, T, Z)] = \Pr_{\delta}[\left\{\begin{matrix} R_S = R, R_S \in_R G_2 \\ w_S = w, w_S \in_R Z_q \\ T_S = T, T_S \in_R G_1, \\ Z_S = Z \end{matrix}\right\}] = 1/q^3$

Proof: At first, R_S, w_S, T_S are chosen from G_2, Z_q, G_1 respectively. As $R_S = R,\ w_S = w,$ $T_S = T$, we have $V_S = V, h_S = h$ by the verification equations defined in section 5.

In the following, we know that:

$$e(Q_{ID_D}, Z) = R \bullet e(sk_{ID_D}, (h+w) \cdot Q_{ID_S})\ .$$

$$= R_S \bullet e(sk_{ID_D}, (h_S + w_S) \cdot Q_{ID_S}) = e(Q_{ID_D}, Z_S)$$

Hence this result implies that $Z_S = Z$.

Claim 2: $\Pr_{\delta'}[(R_D, w_D, T_D, Z_D) = (R, w, T, Z)] = \Pr_{\delta'}[\left\{\begin{matrix} Z_D = Z, Z_D \in_R G_1 \\ R_D = R, R_D \in_R G_2 \\ w_D = w, w_D \in_R Z_q \\ T_D = T \end{matrix}\right\}] = (1/q^3)(1 - 1/q)$

Proof: At first, Z_D, R_D, w_D are chosen from G_1, G_2, Z_q respectively.

As $R_D = R,\ w_D = w, Z_D = Z$, we have the following equations:

$$e(Q_{ID_D}, Z) = e(Q_{ID_D}, Z_D)$$

$$e(Q_{ID_D}, Z) = R \bullet e(sk_{ID_D}, (h+w) \cdot Q_{ID_S})$$

$$= R_D \bullet e(sk_{ID_D}, (h + w_D) \cdot Q_{ID_S})$$

$$e(Q_{ID_D}, Z_D) = R_D \bullet e(sk_{ID_D}, (h_D + w_D) \cdot Q_{ID_S})$$

Hence this result implies that $h = h_D$. Then we know that $\Pr[V_D = V \mid h = h_D] = 1 - 1/q$ due to the fact that $h = G(ID_S, ID_D, R, V, m)$ and the hash function G is assumed to be a random function. So we can assume that $V_D = V$. Finally, we have the following equations:

$$V = e(P, T + w \cdot sk_{ID_D})$$

$$V_D = e(P, T_D + w_D \cdot sk_{ID_D}) = e(P, T_D + w \cdot sk_{ID_D})$$

It is easy to see that $T_D = T$ on condition that $V_D = V$.

The statistical distance between δ and $\delta^/$ is $1/q \approx 1/2^k$. Hence, both distributions of probabilities are indistinguishable for a polynomial time(in k) adversary according to Claim 1 and Claim 2.

Remark: As the distributions are statistically close(i.e., indistinguishable for a polynomial-time adversary), access to simulated signatures will not help the adversary. Hence we will not provide the adversary with simulated signatures when analyzing the unforgeability of our scheme.

The Splitting Lemma [6]: Let $A \subset X \times Y$ such that $\Pr[(x, y) \in A] \ge \gamma$. For any $\alpha < \gamma$, define $B = \{(x, y) \in X \times Y \mid \Pr_{y^/ \in Y}[(x, y^/) \in A] \ge \gamma - \alpha\}$, then $\Pr[B \mid A] \ge \alpha/\gamma$ holds.

Theorem 1: Let the knowledge error k =0. Assume that there is an algorithm F that can make queries to a random oracle G and produce a real signature σ on input a message m with probability at least ε . Let q_G be a bound on the number of queries (R_j, V_j, m_j) made by F to the random oracle G . Then there is a $(4q_G/\varepsilon, 0)$ knowledge extractor K such that $\Pr[x = sk_{ID_S} : x \leftarrow K^{F(\cdot)}(m)] \ge \varepsilon^2/4q_G$.

Proof: Assume that F is a PPT Turing machine with a random tape ω and makes queries to a random oracle G . For a random choice of (ω, G), F can produce a real signature

$\sigma = (R, w, T, Z)$ on input a message m with probability at least ε. Since G is a random oracle, the probability that (R, V, m), where $V = e(P, T + w \cdot sk_{ID_D})$, is not asked to G is at most $1/q$. Hence we can define $Ind(\omega, G)$ to be the index j such that $(R, V, m) = (R_j, V_j, m_j)$. We then define the following sets:

$S = \{(\omega, G) \mid F^G(\omega)$ succeeds and $(1 \le Ind(\omega, G) \le q_G)\}$

$S_i = \{(\omega, G) \mid F^G(\omega)$ succeeds and $(Ind(\omega, G) = i, 1 \le i \le q_G)\}$

We now apply the Splitting Lemma for each $i, 1 \le i \le q_G$. We denote by G_i the restriction of G to queries of index strictly less than i. Let $\alpha = \varepsilon/2q_G$, $\gamma = \varepsilon/q_G$. Since $\Pr[S_i] = \varepsilon/q_G$, it is easy to see that there exists a subset Ω_i of executions by the Splitting Lemma such that:

For any （ω, G）$\in \Omega_i$, and any $G^/$, if $\Pr[(\omega, G^/) \in S_i \mid G_i^/ = G_i] \ge \varepsilon/2q_G$, then $\Pr[\Omega_i \mid S_i] \ge 1/2$, where $G^/$ is another random oracle.

Since all the subsets S_i are disjoint and $\Pr[S_i \mid S]$ is $1/q_G$, we have

$$\Pr_{\omega,G}[\exists(1 \le i \le q_G), (\omega, G) \in \Omega_i \cap S_i \mid S]$$
$$= \sum_{1 \le i \le q_G} \Pr[\Omega_i \cap S_i \mid S] = \sum_{1 \le i \le q_G} \Pr[\Omega_i \mid S_i] \Pr[S_i \mid S] \ge 1/2 .$$

We let β denote the index $Ind(\omega, G)$ corresponding to the successful pair. With probability at least $1/2$, we have $1 \le \beta \le q_G$ and $(\omega, G) \in \Omega_\beta \cap S_\beta$ by the above argument. Hence with probability at least $\varepsilon/2$, F can produce a real signature $\sigma = (R_\beta, w_\beta, T_\beta, Z)$ with such an index β.

In the following, if we replay F with the fixed random tape ω but a randomly chosen oracle $G^/$ such that $G_\beta^/ = G_\beta$, we know $\Pr_{G^/}[(\omega, G^/) \in S_\beta \mid G_\beta^/ = G_\beta] \ge \varepsilon/2q_G$ since $(\omega, G) \in \Omega_\beta \cap S_\beta$. Then

$$\Pr_{G^/}[(\omega, G^/) \in S_\beta \text{ and } h_\beta^/ \ne h_\beta \mid G_\beta^/ = G_\beta]$$

$$\geq \Pr_{G'}[(\omega, G') \in S_\beta \mid G'_\beta = G_\beta] - \Pr_{G'}[h'_\beta = h_\beta] \geq (\varepsilon/2q_G - 1/q) \approx \varepsilon/2q_G \ ,$$

where $h'_\beta = G'(ID_S, ID_D, R_\beta, V_\beta, m_\beta), h_\beta = G(ID_S, ID_D, R_\beta, V_\beta, m_\beta)$.

Then with probability $\varepsilon/2q_G$ we get another success $\sigma' = (R_\beta, w_\beta, T_\beta, Z')$, where $R_\beta = e(P, r_\beta \cdot P)$, $T_\beta = t_\beta \cdot P$, $V_\beta = e(mpk, t_\beta \cdot P + w_\beta \cdot Q_{ID_D})$ and $r_\beta, w_\beta, t_\beta$ are the fixed random coins used by F .

Finally, we have the following equations with probability at least $\varepsilon^2/4q_G$:

$$Z = r_\beta \cdot P + (h_\beta + w_\beta) \cdot sk_{ID_S}$$

$$Z' = r_\beta \cdot P + (h'_\beta + w_\beta) \cdot sk_{ID_S}$$

Hence $sk_{ID_S} = (h'_\beta - h_\beta)^{-1} \cdot (Z' - Z)$.

Theorem 2: Let the knowledge error k =0. Assume that there is an algorithm F that can make queries to a random oracle G and produce a valid simulated signature σ on input a message m with probability at least ε. Let q_G be a bound on the number of queries (R_j, V_j, m_j) made by F to the random oracle G . Then there is a $(4q_G/\varepsilon, 0)$ knowledge extractor K such that $\Pr[x = sk_{ID_D} : x \leftarrow K^{F(\cdot)}(m)] \geq \varepsilon^2/4q_G$.

Proof: Assume that F is a PPT Turing machine with a random tape ω and makes queries to a random oracle G . For a random choice of (ω, G), F can produce a valid simulated signature $\sigma = (R, w, T, Z)$ on input a message m with probability at least ε . Since G is a random oracle, the probability that (R, V, m), where $V = e(P, T + w \cdot sk_{ID_D})$, is not asked to G is at most $1/q$. Hence we can define $Ind(\omega, G)$ to be the index j such that $(R, V, m) = (R_j, V_j, m_j)$. We then define the sets:

$$S = \{(\omega, G) \mid F^G(\omega) \text{ succeeds and } (1 \leq Ind(\omega, G) \leq q_G)\}$$

$$S_i = \{(\omega, G) \mid F^G(\omega) \text{ succeeds and } (Ind(\omega, G) = i, 1 \leq i \leq q_G)\}$$

We now apply the Splitting Lemma for each $i, 1 \le i \le q_G$. We denote by G_i the restriction of G to queries of index strictly less than i. Let $\alpha = \varepsilon / 2q_G$, $\gamma = \varepsilon / q_G$. Since $\Pr[S_i] = \varepsilon / q_G$, it is easy to see that there exists a subset Ω_i of executions by the Splitting Lemma such that:

For any （$(\omega, G) \in \Omega_i$, and any $G^{/}$, $\Pr[(\omega, G^{/}) \in S_i \mid G_i^{/} = G_i] \ge \varepsilon / 2q_G$, then $\Pr[\Omega_i \mid S_i] \ge 1/2$, where $G^{/}$ is another random oracle.

Since all the subsets S_i are disjoint and $\Pr[S_i \mid S]$ is $1/q_G$, we have

$$\Pr_{\omega, G}[\exists (1 \le i \le q_G), (\omega, G) \in \Omega_i \bigcap S_i \mid S]$$

$$= \sum_{1 \le i \le q_G} \Pr[\Omega_i \bigcap S_i \mid S] = \sum_{1 \le i \le q_G} \Pr[\Omega_i \mid S_i] \Pr[S_i \mid S] \ge 1/2 .$$

We let β denote the index $Ind(\omega, G)$ corresponding to the successful pair. With probability at least $1/2$, we have $1 \le \beta \le q_G$ and $(\omega, G) \in \Omega_\beta \bigcap S_\beta$. Hence with probability at least $\varepsilon / 2$, F can produce a valid simulated signature $\sigma = (R_\beta, w_\beta, T_\beta, Z_\beta)$ with such an index β.

In the following, if we replay F with the fixed random tape ω but randomly chosen oracle $G^{/}$ such that $G_\beta^{/} = G_\beta$, we know $\Pr_{G^{/}}[(\omega, G^{/}) \in S_\beta \mid G_\beta^{/} = G_\beta] \ge \varepsilon / 2q_G$ since $(\omega, G) \in \Omega_\beta \bigcap S_\beta$. Then

$$\Pr_{G^{/}}[(\omega, G^{/}) \in S_\beta \text{ and } h_\beta^{/} \ne h_\beta \mid G_\beta^{/} = G_\beta]$$

$$\ge \Pr_{G^{/}}[(\omega, G^{/}) \in S_\beta \mid G_\beta^{/} = G_\beta] - \Pr_{G^{/}}[h_\beta^{/} = h_\beta] \ge (\varepsilon / 2q_G - 1/q) \approx \varepsilon / 2q_G ,$$

where $h_\beta^{/} = G^{/}(ID_S, ID_D, R_\beta, V_\beta, m_\beta), h_\beta = G(ID_S, ID_D, R_\beta, V_\beta, m_\beta)$.

Then with probability $\varepsilon / 2q_G$ we get another success $\sigma^{/} = (R_\beta, w_\beta^{/}, T_\beta^{/}, Z_\beta)$, where $R_\beta = e(Q_{ID_D}, Z_\beta) \bullet e(sk_{ID_D}, -\lambda_\beta \cdot Q_{ID_S})$ and the fixed random coins used by F in this case are $\alpha_\beta, \lambda_\beta, Z_\beta$ since $\sigma^{/}$ is a valid simulated signature. Note that $w_\beta = (\lambda_\beta - h_\beta) \bmod q$, $w_\beta^{/} = (\lambda_\beta - h_\beta^{/}) \bmod q$. Hence $w_\beta \ne w_\beta^{/}$. This result also implies that $T_\beta \ne T_\beta^{/}$.

Finally, we have the following equations with probability at least $\varepsilon^2/4q_G$:

$T_\beta = (\alpha_\beta \cdot P - w_\beta \cdot sk_{ID_D})$

$T_\beta^/ = (\alpha_\beta \cdot P - w_\beta^/ \cdot sk_{ID_D})$

Hence $sk_{ID_D} = (w_\beta - w_\beta^/)^{-1} \cdot (T_\beta^/ - T_\beta)$.

Theorem 3: Assume (G_1, G_2) satisfies the GBDH assumption. Suppose there is a polynomial-time adversary A who makes at most q_k **UserKey** queries, q_s **Sign** queries and q_v **Verf** queries can existentially forge a signature in our scheme with non-negligible success probability ε in time at most t.Then there is an algorithm B that solves the GBDH problem in (G_1, G_2) with probability:

$$\varepsilon^/ > (1 - 2/(a+2))^a (2/(a+2)^2))\varepsilon \text{, where } a = q_k + q_s + q_v .$$

Proof: Algorithm B is given as input a tuple $(P, a \cdot P, b \cdot P, c \cdot P)$, where P is the generator of the group G_1 with prime order q, $a, b, c \in Z_q$. Then B works by interacting with the adversary A as follows.

The system parameters are **params=** $< (G_1, +), (G_2, \bullet), e, q, P, G, H >$, where G, H are the random oracles controlled by B.

At first, B initializes mpk with $c \cdot P$ and picks n identities $X = \{ID_1, \cdots, ID_n\}$. Then B randomly picks two identities $ID_S, ID_D \in X$. During the simulation, B can answer A's queries as follows:

H **Queries:** B maintains a list $H^{list} = \{< ID, l, H(ID) >\}$. On input an identity $ID \in X$ chosen by A, if the queried identity ID appears in H_1^{list}, B returns the previously assigned value. Otherwise, B performs as follows:

(1) $ID \notin \{ID_S, ID_D\}$, B picks a random $l \in Z_q$ and responds to A with $H(ID) = l \cdot P$.

(2) $ID = ID_S$, B responds to A with $H(ID) = a \cdot P$.

(3) $ID = ID_D$, B responds to A with $H(ID) = b \cdot P$.

G **Queries:** B maintains a list $G^{list} = \{< (ID_i, ID_j, R, V, m), g\_hash\}$. On input (ID_i, ID_j, R, V, m) chosen by A, if the queried tuple appears in G^{list} , B returns the previously assigned value. Otherwise, B picks a random $k \in Z_q$ and responds to A with $g\_hash = k$.

UserKey queries: B maintains a list $L = \{< ID, sk_{ID} >\}$. On input an identity $ID \in X$ chosen by A , if the queried identity ID appears in L , B returns the previously assigned value. Otherwise, B performs as follows:

(1) $ID \notin \{ID_S, ID_D\}$, B looks up H^{list} to extract a tuple $< ID, l, H(ID) >$ and responds to A with $sk_{ID} = l \cdot mpk$. Then $Corr \leftarrow Corr \bigcup \{ID\}$.

(2) $ID \in \{ID_S, ID_D\}$, B returns $\perp$ and aborts.

Sign queries: On input a signer's identity ID_i , a verifier's identity ID_j and a message m adaptively chosen by A, if $ID_i \neq ID_j$, B performs as follows:

(1) $|\{ID_i, ID_j\} \bigcap \{ID_S, ID_D\}| \leq 1$: If $ID_i \notin \{ID_S, ID_D\}$, B looks up L to extract sk_{ID_i} and returns $\sigma_{DV} \leftarrow$ S**DV_Sign** (sk_{ID_i}, ID_j, m) . Otherwise, B looks up L to extract sk_{ID_j} and returns $\overline{\sigma_{DV}} \leftarrow$ **SDV_Sim** (sk_{ID_j}, ID_i, m) .

(2) $\{ID_i, ID_j\} = \{ID_S, ID_D\}$: B returns $\perp$ and aborts.

Verf queries: Given a signer's identity ID_i , a verifier's identity ID_j , and a message/signature pair $(m, \sigma = (R, w, T, Z))$ provided by A , $ID_i \neq ID_j$, if $ID_j \notin \{ID_S, ID_D\}$, B returns the result of running SDV_Verf $(sk_{ID_j}, ID_i, m, \sigma)$. Otherwise, B returns $\perp$ and aborts.

Finally, if B does not abort during the simulation, A will output a signer's identity

ID_i , a verifier's identity ID_j , $ID_i \neq ID_j$, and a message/signature pair $(m^*, \sigma^* = (R^*, w^*, T^*, Z^*))$. If $\{ID_i, ID_j\} \neq \{ID_S, ID_D\}$, B returns $\perp$ and aborts. Otherwise, if the forgery output by the adversary A is successful, the probability that A does not issue a G query is at most $1/q$. Hence we know that $(ID_i, ID_j, R^*, V^*, m^*)$ is already in G^{list} with probability at least $1-1/q$ in this case. For the sake of simplicity, we only consider the case $ID_i = ID_S, ID_j = ID_D$. The other case $ID_i = ID_D, ID_j = ID_S$ can be analyzed similarly.

So we have $h^* = G(ID_S, ID_D, R^*, V^*, m^*)$, $Q_{ID_i} = Q_{ID_S} = H(ID_S) = a \cdot P$, $Q_{ID_j} = Q_{ID_D} = H(ID_D)$ = $b \cdot P$, sk_{ID_D} = $(cb) \cdot P$. Let ξ = $e(b \cdot P, Z^*)$. If $O_{DBDH}(a \cdot P, b \cdot P, c \cdot P, (\xi/R^*)^{(h^*+w^*)^{-1}})$ =1, then B returns $(\xi/R^*)^{(h^*+w^*)^{-1}}$. The reason is that the following equations hold:

$$e(Q_{ID_D}, Z^*) = R^* \bullet e(sk_{ID_D}, (h^* + w^*) \cdot Q_{ID_S}),$$

$$e(P, P)^{abc} = (\xi/R^*)^{(h^*+w^*)^{-1}}.$$

Now it remains to analyze the probability of B not aborting. B aborts if the following events happens:

E_1: B aborts when answering **UserKey** queries.

E_2: B aborts when answering **Sign** queries.

E_3: B aborts when answering **Verf** queries.

E_4: A outputs ID_i and ID_j such that $\{ID_i, ID_j\} \neq \{ID_S, ID_D\}$.

It is easy to see that $\Pr[E_1]$ = $2/n$, $\Pr[E_2]$ = $2/n(n-1)$, $\Pr[E_3] = 2/n$, $\Pr[E_4] = 1-(2/n(n-1))$. Hence the success probability $\varepsilon^/$ of B can be estimated as follows:

$$\varepsilon^/ = (1-2/n)^{q_k+q_v}(1-2/n(n-1))^{q_s}(2/n(n-1))\varepsilon$$

$$> (1-2/n)^{q_k+q_v+q_s}(2/n^2))\varepsilon$$

Let $a = q_k + q_s + q_v$, $\eta(n) = (1 - 2/n)^a (2/n^2))$. Then $\eta(n)$ is maximized at $n = (a+2)$. Hence for large a, $\varepsilon^/ > \dfrac{2\varepsilon}{\exp(2)\cdot a^2}$.

The running time of B can be calculated as

$$t + ((n-2) + 5q_s + 2q_v)t_m + (2q_s + 3q_v)t_p + q_k O(n)$$

where t_m is the time to compute a scalar multiplication in G_1 and t_p is the time to compute a pairing operation.

Theorem 4: Our IDSDVS scheme is non-transferable against a polynomial-time distinguisher D who makes at most q_k **UserKey** queries, q_s **Sign** queries, q_{sim} **Sim** queries and q_v **Verf** queries.

Proof: According to Lemma 1, the distributions of δ (the real signatures), $\delta^/$ (the simulated signatures) are statistically close. Consequently, it is infeasible for a polynomial-time distinguisher D to distinguish the simulated signatures from the real signatures. Hence our IDSDVS scheme is non-transferable.

7. Performance Analysis

In this section, we evaluate the performance of our scheme and other related schemes proposed in [3, 4, 9] in terms of the signature length and computational cost. In table 1, Mu(G) denotes a multiplication operation in group G. Exp, Pair and Hash denote an exponentiation operation, a pairing operation and a hash operation respectively. Although the scheme of [3] is more efficient than other schemes, no formal security analysis is presented for it. In addition, our scheme is proven to be non-delegatable.

Table 1. Performance comparison with other related schemes

Scheme	Signature length	Signing cost	Verification cost	Non-delegatable
Zhang-Mao scheme [9]	$3\|G_1\|$	$4Mu(G_1)$+1Hash+ 1Inv+1Mu(Z_q)	3P+2Mu(G_2)+ 1Hash	No

Kang et al.’s scheme [4]	$2\|G_1\|$	$2P+3Mu(G_1)+1Hash$	$1P+1Mu(G_2)+1Mu(G_1)$	No
Kang et al.’s scheme [3]	$2\|G_1\|$	$2Mu(G_1)+1Hash+1P$	$1Hash+1P$	No
Our scheme	$3\|G_1\|+1\|Z_q\|$	$5Mu(G_1)+1Hash+2P$	$3P+2Mu(G_1)+1Mu(G_2)+1Hash$	Yes

8.Conclusion

Some previously proposed IDSDVS schemes [4, 9] are vulnerable to the attack described in [3]. The essence of this attack is that the schemes of [4, 9] are delegatable. In this paper, a security model for non-delegatable IDSDVS schemes is established. Then a novel non-delegatable IDSDVS scheme based on pairing is presented. In the following, we provide security proofs to show that our scheme is non-delegatable, non-transferable and unforgeable under the Gap Bilinear Diffie-Hellman assumption. However，the security of our scheme is not tight as the reduction uses the Forking Lemma. It remains to be seen if there exists a non-delegatable IDSDVS scheme that enjoys a tight reduction without using the Forking Lemma. Finally, we compare the efficiency with other related IDSDVS schemes.

References

[1] O. Goldreich, “Foundations of Cryptography: Basic Tools”, Cambridge University Press, 2001.

[2] M. Jacobsson, K. Sako, R, Impagliazzo, "Designated verifier proofs and their applications", in EUROCRYPT 1996, LNCS 1070, 1996, pp.143-154.

[3] B. Kang, C. Boyd, E. Dawson, “Identity-based strong designated verifier signature schemes: Attacks and new construction”, Computers and Electrical Engineering, 2008, (to be published).

[4] B. Kang, C. Boyd, E. Dawson, “A novel identity-based strong designated verifier signature scheme”, The Journal of System and Software, 2008, (to be published).

[5] H. Lipmaa, G. Wang, and F. Bao, “Designated verifier signature schemes: attacks, new security notions and a new construction”, ICALP 2005, LNCS 3580, 2005, pp.459-471.

[6] D. Pointcheval, and J. Stern, “Security arguments for digital signatures and blind signatures”, Journal of Cryptology, 2000(13), pp.361-396.

[7] S. Saeednia, S. Kremer, and O.Markowitch, “An efficient designated verifier signature scheme”, ICISC 2003, LNCS 2971, 2003, pp.40-54.

[8] W. Susilo, F. Zhang, and Y. Mu, “Identity-based strong designated verifier signature schemes”, ACISP 2004, LNCS 3108, 2004, pp.313-324.

[9] J. Zhang, and J. Mao, “A novel ID-based strong designated verifier signature scheme”, Information Science, 178(2008), pp.766-773.

Corresponding author: Bin Wang

Address: P.O.Box 153#
Information Engineering College
No.36 Middle JiangYang Road
Yangzhou University, Yangzhou City, Jiangsu Province, Peoples Republic of China

Postal code: 225009

E-mail:jxbin76@yahoo.cn