JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 28, 771-785 (2012)

Short Paper

An Efficient Strong Designated Verifier Proxy Signature Scheme for Electronic Commerce

HAN-YU LIN, TZONG-SUN WU+ AND SHIH-KUN HUANG
Department of Computer Science
National Chiao Tung University
Hsinchu, 300 Taiwan
+Department of Computer Science and Engineering
National Taiwan Ocean University
Keelung, 202 Taiwan

A strong designated verifier signature (SDVS) scheme only allows a designated verifier to validate signer's signatures for ensuring confidentiality. At the same time, the designated verifier can not transfer the signature to any third party, since he can also generate another computationally indistinguishable SDVS, which is referred to as non-transferability. A proxy signature scheme is a special type of digital signature schemes, which enables an authorized proxy signer to create a valid proxy signature on behalf of the original one. The resulted proxy signature is publicly verifiable by anyone. In this paper, we elaborate on the merits of SDVS schemes and proxy signature schemes to propose an efficient strong designated verifier proxy signature (SDVPS) scheme in which only a designated verifier can be convinced of the proxy signer's identity. The proposed scheme has crucial benefits in organizational operations and electronic commerce. Compared with related schemes, ours has not only shorter signature length, but also lower computational costs. Moreover, the security requirement of unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA) is proved in the random oracle model.

Keywords: designated verifier, digital signature, proxy, discrete logarithms, public key system

1. INTRODUCTION

In a digitalized world, digital signatures [1, 2] serve the same function as traditional handwritten signatures, which provide the properties of integrity, authenticity [3] and non-repudiation [4]. In 1996, Mambo *et al*. [5, 6] introduced proxy signature schemes in which a properly delegated proxy signer has the ability to sign messages on behalf of an original signer such that anyone can verify the corresponding proxy signature with proxy signer's public key. It thus can be seen that proxy signature schemes are applicable in electronic commerce and mobile agents, *etc*. When it comes to proxy delegation, it can be categorized into four different kinds: full delegation, partial delegation, delegation by warrant [7, 8] and partial delegation with warrant [9]. In full delegation, an original signer directly gives a proxy signer his private key as the proxy signing key. Consequently, all the (proxy) sig-

Received July 9, 2010; revised September 23 & November 2, 2010; accepted November 28, 2010.
Communicated by Tzong-Chen Wu.

natures are generated with the same private key. It is difficult for any third party to identify the actual signer from a given signature. In partial delegation, an original signer further derives another proxy signing key with his own private key. Even with the knowledge of the proxy signing key, it is computationally infeasible for any polynomial-time adversary to compute original signer's private key. However, a malicious original signer can easily impersonate the proxy signer to create valid proxy signatures. In delegation by warrant, a warrant composed of some proxy information, *e.g*., the proxy period and the identities of original and proxy signers is delivered to the proxy signer who thus has to spend more computational efforts for certifying the warrant. To obtain better efficiency, partial delegation with warrant is a better alternative, since certifying the warrant can be combined with subsequent procedures.

In the same year, Jakobsson *et al*. [10] proposed another variant of digital signature scheme called designated verifier signature (DVS) scheme. In such a scheme, only a designated verifier can be convinced of the validity of received signature with respect to some signer. The designated verifier can not transfer the signature to any third party, because he is also capable of creating a computationally indistinguishable transcript intended for himself, which is referred to as non-transferability. One can see that DVS schemes are suitable in the applications such as electronic voting [11, 12] in which the non-repudiation property is not desirable. Yet, in 2003, Wang [13] pointed out that Jakobsson *et al.*'s scheme is insecure, as a malicious signer can easily cheat the designated verifier. Later, Saeednia *et al*. [14] addressed a strong designated verifier signature (SDVS) scheme which only allows a designated verifier to validate received signatures in relation to some signer. Since the designated verifier's private key is a vital parameter for performing signature validation equation, anyone without such information can not verify the signature. So far, lots of SDVS schemes [15-22] have been proposed and extensively studied.

Consider the situation of on-line auction that a famous handicraft designer asks an auction manager to sale his precious product for charities. The winning bidder will obtain the product along with a signed receipt which can be used to prove the handicraft's legal origin. The auction manager also hopes that only the winning bidder is able to verify the signed receipt such that he can not illegally resell the precious handicraft to black market for more profits. Another commonly seen application is confidential contract signing. A company can authorize one legal agent to sign business contracts such that only an intended company is capable of validating the contract. To solve above application requirements, in 2003 and 2005, Dai *et al*. [23] and Wang [24] separately proposed designated verifier proxy signature schemes in which a proxy signer can generate a valid proxy DVS on behalf of an original singer such that only a designated verifier is able to verify it. Nevertheless, their schemes are inefficient in terms of computational efforts and communicational overheads. In this paper, we propose an efficient strong designated verifier proxy signature (SDVPS) scheme with provable security in the random oracle model. Compared with related works, our scheme not only has shorter signature length, but also earns more computational efficiency.

The rest of this paper is organized as follows. Section 2 states some preliminaries. We introduce the proposed SDVPS scheme in section 3. The security proof and some comparisons are detailed in section 4. Finally, a conclusion is made in section 5.

2. PRELIMINARIES

In this section, we first define used notations as Table 1 and then briefly review some security notions along with the computational assumptions.

Discrete Logarithm Problem (DLP) [25] Let p and q be two large primes satisfying $q \mid p - 1$, and g a generator of order q over GF(p). The discrete logarithm problem is, given an instance (y, p, q, g), where $y = g^x \bmod p$ for some $x \in Z_q$, to derive x.

Discrete Logarithm (DL) Assumption [25] A probabilistic polynomial-time algorithm $\mathcal{B}$ is said to (t, ε)-break the DLP if given a DLP instance (y, p, q, g) where $y = g^x \bmod p$ for some $x \in Z_q$, $\mathcal{B}$ can derive x with probability ε after running at most t steps. The probability is taken over the uniformly and independently chosen instance and over the random bits consumed by $\mathcal{B}$.

Definition 1 The (t, ε)-DL assumption holds if there is no probabilistic polynomial-time adversary that can (t, ε)-break the DLP.

Table 1. The used notations.

Z_p	integers modulo p
Z^*_p	multiplicative group of integers modulo p
GF(p)	Galois field of p elements
$X \in Z_p$	element x in set Z_p
$X \in_R Z_p$	element x is a random integer in set Z_p
$x \leftarrow Z_p$	sampling element x uniformly in set Z_p
$a \bmod b$	modulo operation: reminder of a divided by b
$a \mid b$	integer b is divisible by integer a
$a \mid\mid b$	concatenation of a and b
$\lvert x \rvert$	bit-length of integer x, also absolute value of x
$\log_b x$	logarithm to base b of x
$\neg$	logical operation NOT
$\wedge$	logical operation AND
$\vee$	logical operation OR
$\forall$	for all
Pr[E]	probability of event E occurring

3. THE PROPOSED SCHEME

In this section, we first address involved parties and algorithms of our proposed scheme and then give a concrete construction.

3.1 Involved Parties

An SDVPS scheme has three involved parties: an original signer, a proxy signer and a

designated verifier. Each one is a probabilistic polynomial-time Turing machine (PPTM). The original signer will compute and transmit a proxy credential to the proxy signer. The proxy signer is responsible for generating an SDVPS intended for the designated verifier on behalf of the original signer. Finally, the designated verifier validates the proxy signature with his private key. An SDVPS scheme is correct if the proxy signer can generate a valid SDVPS which can only be verified by the designated verifier.

3.2 Algorithms

The proposed SDVPS scheme consists of the following algorithms:

- **Setup:** Taking as input 1^k where k is a security parameter, the algorithm generates system's public parameters *params*.
- **Proxy-Credential-Generation (PCG):** The PCG algorithm takes as input system para- meters *params* and the private key of original signer. It outputs a corresponding proxy credential.
- **Proxy-Signature-Generation (PSG):** The PSG algorithm takes as input system parameters *params*, a proxy credential, a message, the public key of designated verifier and the private key of proxy signer. It generates an SDVPS δ.
- **Proxy-Signature-Verification (PSV):** The PSV algorithm takes as input system parameters *params*, a message m, an SDVPS δ, the private key of designated verifier and the public keys of original and proxy signers. It outputs **True** if δ is a valid SDVPS for m. Otherwise, an error symbol $\perp$ is returned as a result.

3.3 Concrete Construction

We demonstrate the proposed SDVPS scheme over a finite field. Details are described below:

- **Setup:** Taking as input 1^k, the system authority (SA) selects two large primes p and q where $|q| = k$ and $q \mid (p-1)$. Let g be a generator of order q and h_1: $\{0, 1\}^k \times Z_q \rightarrow Z_q$, h_2: $\{0, 1\}^* \times Z_q \rightarrow Z_q$ and h_3: $Z_q \rightarrow Z_q$ collision resistant hash functions. The system publishes public parameters *params* = $\{p, q, g, h_1, h_2, h_3\}$. Each user U_i chooses his private key $x_i \in Z_q$ and computes the public key as $y_i = g^{x_i}$.
- **Proxy-Credential-Generation (PCG):** Let U_o be an original user delegating his signing power to a proxy signer U_p. U_o first chooses $d \in_R Z_q$ to compute

$$T = (g^d \bmod p) \bmod q, \tag{1}$$
$$\sigma = d - x_o h_1(m_w, T) \bmod q, \tag{2}$$

where m_w is a warrant consisting of the identifiers of original and proxy signers, the delegation duration and so on. (σ, m_w, T) is then sent to U_p. Upon receiving (σ, m_w, T), U_p computes Z as Eq. (3) and performs Eq. (4) to check its validity.

$$Z = y_o^{h_1(m_w, T)} \bmod p, \tag{3}$$
$$T = g^{\sigma} Z \bmod p \pmod q. \tag{4}$$

If it does not hold, (σ, m_w, T) is requested to be sent again. We demonstrate that the verification of Eq. (4) works correctly. From the right-hand side of Eq. (4), we have

$$\begin{aligned} & g^{\sigma}Z \\ &= g^{\sigma}y_o^{h_1(m_w,T)} && \text{by Eq. (3)} \\ &= g^{d-x_o h_1(m_w,T)}y_o^{h_1(m_w,T)} && \text{by Eq. (2)} \\ &= g^{d} \\ &= T(\text{mod } q) && \text{by Eq. (1)} \end{aligned}$$

which leads to the left-hand side of Eq. (4).

– **Proxy-Signature-Generation (PSG):** For signing a message $m \in_R \{0, 1\}^*$ on behalf of the original signer U_o, U_p chooses $w \in_R Z_q$ to compute

$$s_1 = h_3((y_v^w \bmod p) \bmod q), \quad (5)$$
$$s_2 = w - (x_p + \sigma)h_2(m, T) \bmod q, \quad (6)$$

and then delivers (m, m_w) along with the SDVPS $\delta = (s_1, s_2, T)$ to a designated recipient U_v.

– **Proxy-Signature-Verification (PSV):** Upon receiving (m, m_w) and δ, U_v first computes (R_1, R_2) as follows:

$$R_1 = y_v^{s_2} \bmod p, \quad (7)$$
$$R_2 = (Ty_p y_o^{-h_1(m_w,T)})^{x_v h_2(m,T)} \bmod p. \quad (8)$$

U_v then verifies the proxy signature by checking if

$$s_1 = h_3((R_1R_2 \bmod p) \bmod q). \quad (9)$$

If it holds, the SDVPS $\delta = (s_1, s_2, T)$ for m is valid. We show that the verification of Eq. (9) works correctly. From the right-hand side of Eq. (9), we have

$$\begin{aligned} & R_1R_2 \\ &= y_v^{s_2}(Ty_p y_o^{-h_1(m_w,T)})^{x_v h_2(m,T)} && \text{by Eqs. (7) and (8)} \\ &= y_v^{s_2}(y_p g^{\sigma})^{x_v h_2(m,T)} && \text{by Eq. (4)} \\ &= y_v^{s_2}(y_v)^{(x_p+\sigma)h_2(m,T)} \\ &= y_v^{s_2+(x_p+\sigma)h_2(m,T)} \\ &= y_v^{w}(\text{mod } p) && \text{by Eq. (6)} \end{aligned}$$

which implies $h_3((R_1R_2 \bmod p) \bmod q) = h_3((y_v^w \bmod p) \bmod q) = s_1$. by Eq. (5)

4. SECURITY PROOF AND COMPARISON

In this section, we first define the security model of our proposed SDVPS scheme and prove it in the random oracle model. Then some comparisons with related schemes are made.

4.1 Security Model

The essential security requirements of the proposed SDVPS scheme are non-transferability and unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA). We define these notions as follows.

Definition 2 (Unforgeability) An SDVPS scheme is said to achieve the security requirement of unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA) if there is no probabilistic polynomial-time adversary $\mathcal{A}$ with non- negligible advantage in the following game played with a challenger $\mathcal{B}$: (Note that the challenger $\mathcal{B}$ is responsible for answering queries of the adversary $\mathcal{A}$ who attempts to forge a valid SDVPS of the proposed scheme.)

Setup: $\mathcal{B}$ first runs Setup(1^k) algorithm and sends system's public parameters *params* to the adversary $\mathcal{A}$.

Phase 1: The adversary $\mathcal{A}$ can issue several queries adaptively, *i.e*., each query might be based on the result of previous queries:
– *Proxy-Credential-Generation* (*PCG*) *queries:* $\mathcal{A}$ issues a PCG query with respect to a proxy signer. $\mathcal{B}$ returns a corresponding proxy credential.
– *Proxy-Signature-Generation* (*PSG*) *queries:* $\mathcal{A}$ chooses a message *m* and then gives it to $\mathcal{B}$ who returns an SDVPS δ to $\mathcal{A}$.
– *Proxy-Signature-Verification* (*PSV*) *queries:* $\mathcal{A}$ gives $\mathcal{B}$ a message *m* and an SDVPS δ. If δ is a valid SDVPS for *m*, $\mathcal{B}$ runs **True**. Otherwise, an error symbol $\perp$ is returned as a result.

Forgery: Finally, $\mathcal{A}$ produces a new pair (m^*, δ^*) which is not outputted by any PSG query. The adversary $\mathcal{A}$ wins if δ^* is a valid SDVPS for m^*.

Definition 3 (Non-Transferability) An SDVPS scheme is said to achieve the security requirement of non-transferability if a designated verifier can simulate a computationally indistinguishable transcript intended for himself with his private key.

Definition 4 (Strong Privacy of Signer's Identity) An SDVPS scheme satisfies the security requirement of strong privacy of signer's identity if there is no probabilistic polynomial-time adversary having the ability to determine the identity of signer for an intercepted SDVPS by performing the signature verification process before the SDVPS has been received by the designated verifier.

4.2 Security Proof

We prove that the proposed scheme achieves the essential security requirements defined above. As for the EF-CMA security, if we directly apply the proof techniques of Forking Lemma addressed by Pointcheval and Stern [26] to prove our scheme, we can also obtain the following results.

The Forking Lemma In the random oracle mode, let $\mathcal{A}$ be a probabilistic polynomial-time Turing machine whose input only consists of public data. We denote respectively by N_1 and N_2 the number of queries that $\mathcal{A}$ can ask to the random oracle and the number of queries that $\mathcal{A}$ can ask to the signer. Assume that, within a time bound R, $\mathcal{A}$ produces, with probability $\varepsilon \geq 10(N_2 + 1)(N_2 + N_1)/2^k$, a valid signature $(m, \Sigma_1, H, \Sigma_2)$ where $\Sigma_1 = (s_1, m_w, T)$, $H = (h_1(m_w, T), h_2(m, T))$ and $\Sigma_2 = s_2$. If the triples (Σ_1, H, Σ_2) can be simulated without knowing the private key with an indistinguishable distribution probability, then there is another machine which has control over the machine obtained from $\mathcal{A}$ replacing interaction with the signer by simulation and produces two valid signatures $(m, \Sigma_1, H, \Sigma_2)$ and $(m, \Sigma_1, H', \Sigma_2')$ such that $h_2(m, T) \neq h_2'(m, T)$ in the expected time $R' \leq 120686R/\varepsilon$.

Concretely speaking, in our scheme, we can first obtain two equations:

$$s_2 = w - (x_p + \sigma)h_2(m, T) \bmod q,$$
$$s_2' = w - (x_p + \sigma)h_2'(m, T) \bmod q,$$

and then compute the private key x_p as

$$x_p = ((s_2 - s_2') + \sigma(h_2(m, T) - h_2'(m, T)))/(h_2'(m, T) - h_2(m, T)).$$

Nevertheless, to show the tight relation between the security of our SDVPS scheme and the hardness of DLP, we have to present another more detailed security proof and advantage analyses as Theorem 1.

Theorem 1 The proposed SDVPS scheme is $(t, q_{h_1}, q_{h_2}, q_{PCG}, q_{PSG}, q_{PSV}, \varepsilon)$-secure against existential forgery on adaptive chosen-message attacks (*EF-CMA*) in the random oracle model if there is no probabilistic polynomial-time adversary that can (t', ε')-break the DLP, where

$$\varepsilon' \geq 4^{-1}(\varepsilon - 2^{-k})^3(q_{h_2}^{-1}),$$
$$t' \approx t + t_\lambda(4q_{PCG} + 6q_{PSG} + 6q_{PSV}).$$

Here t_λ is the time for performing a modular exponentiation over a finite field.

Proof: Fig. 1 depicts the proof structure of this theorem. Suppose that a probabilistic polynomial-time adversary $\mathcal{A}$ can $(t, q_{h_1}, q_{h_2}, q_{PCG}, q_{PSG}, q_{PSV}, \varepsilon)$-break the proposed SDVPS scheme with non-negligible advantage ε under adaptive chosen message attacks after running at most t steps and making at most q_{h_i} h_i random oracles (for i = 1 and 2), q_{PCG} PCG, q_{PSG} PSG and q_{PSV} PSV queries. Then we can construct another algorithm $\mathcal{B}$ that can (t', ε')-break the DLP by taking $\mathcal{A}$ as a subroutine. Let all involved parties and notations be defined the same as those in section 3.3. The objective of $\mathcal{B}$ is to obtain α (= $\log_g C$) by taking $(p, q, g, C = g^\alpha \bmod p)$ as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs Setup(1^k) algorithm to obtain system's public parameters $params = \{p, q, g\}$ and comes up with a random tape composed of a long sequence of random bits. Then $\mathcal{B}$ chooses $r \in_R Z_q$ to set $y_v' = g^r \bmod p$ and $y_p' = C$. After that, $\mathcal{B}$ simu-

lates two runs of SDVPS scheme to the adversary $\mathcal{A}$ on input $\{p, q, g, y_v', y_p', y_o\}$ and the random tape.

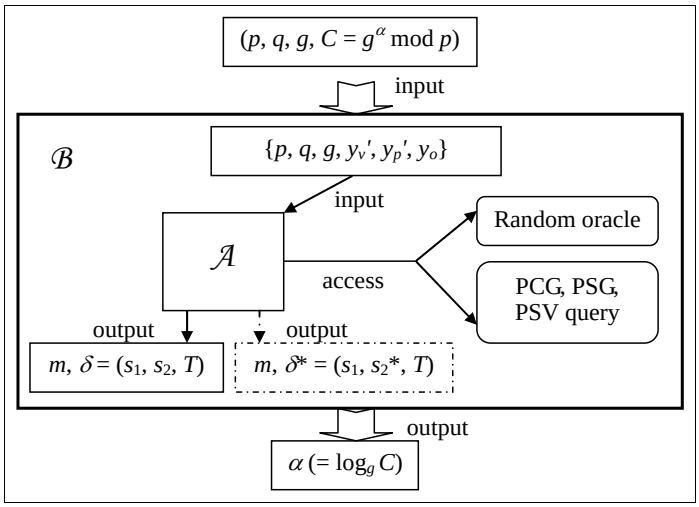

Fig. 1. The proof structure of Theorem 1.

oracle O-**Sim_**h_1(m_w, T)
1: for i = 0 to $q_{h_1} - 1$
2: if (Q_h_1[i][0] = m_w) and (Q_h_1[i][1] = T) then // It is an old query.
3: exit for;
4: else if (Q_h_1[i][0] = null) then // It is a new query.
5: **insert**(Q_h_1, (m_w, T));
6: A_h_1[i] ← $v_1 \in_R Z_q$;
7: exit for;
8: end if
9: next i
10: return A_h_1[i];

Fig. 2. Algorithm of the simulated random oracle O-**Sim_**h_1.

oracle O-**Sim_**h_2(m, T)
1: for i = 0 to $q_{h_2} - 1$
2: if (Q_h_2[i][0] = m) and (Q_h_2[i][1] = T) then // It is an old query.
3: exit for;
4: else if (Q_h_2[i][0] = null) then // It is a new query.
5: **insert**(Q_h_2, (m, T));
6: A_h_2[i] ← $v_2 \in_R Z_q$;
7: exit for;
8: end if
9: next i
10: return A_h_2[i];

Fig. 3. Algorithm of the simulated random oracle O-**Sim_**h_2.

oracle $\mathcal{O}$**-Sim_*PCG***(m_w)
1: do
2: Choose $\sigma, v_1 \in_R Z_q$;
3: Compute $(T = g^{\sigma} y_o^{v_1} \bmod p) \bmod q$;
4: while (**check**(Q_h_1, (m_w, T)) = true)
5: **insert**(Q_h_1, (m_w, T)); **insert**(A_h_1, v_1); // define $h_1(m_w, T) = v_1$
6: return (σ, T);

Fig. 4. Algorithm of the simulated PCG oracle $\mathcal{O}$**-Sim_*PCG***.

oracle $\mathcal{O}$**-Sim_*PSG***(m)
1: Choose $s_2 \in_R Z_q$ and a proper m_w;
2: $(\sigma, T) \leftarrow \mathcal{O}$**-Sim_*PCG***$(m_w)$;
3: $v_1 \leftarrow \mathcal{O}$**-Sim_$h_1$**$(m_w, T)$; $v_2 \leftarrow \mathcal{O}$**-Sim_$h_2$**$(m, T)$;
4: Compute $R_1 = g^{rs_2} \bmod p$; $R_2 = (Cg^{\sigma})^{rv_2} \bmod p$; $s_1 = h_3((R_1 R_2 \bmod p) \bmod q)$;
5: return $\delta = (s_1, s_2, T)$ and m_w;

Fig. 5. Algorithm of the simulated PSG oracle $\mathcal{O}$**-Sim_*PSG***.

oracle $\mathcal{O}$**-Sim_*PSV***(m, δ, m_w)
1: $v_1 \leftarrow \mathcal{O}$**-Sim_$h_1$**$(m_w, T)$; $Z = y_o^{v_1} \bmod p$; $v_2 \leftarrow \mathcal{O}$**-Sim_$h_2$**$(m, T)$;
2: Compute $R_1 = g^{rs_2}$; $R_2 = (CTZ^{-1})^{rv_2}$; $s_1^* = h_3((R_1 R_2 \bmod p) \bmod q)$;
3: if $(s_1^* = s_1)$ then
4: return **True**;
5: else
6: return $\perp$;
7: end if

Fig. 6. Algorithm of the simulated PSV oracle $\mathcal{O}$**-Sim_*PSV***.

Phase 1: $\mathcal{A}$ makes the following queries adaptively:

– h_1 *oracle:* When $\mathcal{A}$ queries an h_1 oracle of $h_1(m_w, T)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim_h_1**(m_w, T). The simulated random oracle $\mathcal{O}$**-Sim_h_1** operates as Fig. 2. Note that the function **insert**(N, b) will insert the value b into the array N.

– h_2 *oracle:* When $\mathcal{A}$ queries an h_2 oracle of $h_2(m, T)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim_h_2**(m, T). The simulated random oracle $\mathcal{O}$**-Sim_h_2** operates as Fig. 3.

– *PCG queries:* When $\mathcal{A}$ makes a PCG query, $\mathcal{B}$ chooses a proper m_w and then returns $(m_w, \mathcal{O}$**-Sim_*PCG***$(m_w))$ as the result. The simulated PCG oracle $\mathcal{O}$**-Sim_*PCG*** operates as Fig. 4. Note that the function **check**(N, b) will return a Boolean value depending on whether the value b is stored in the array N.

– *PSG queries:* When $\mathcal{A}$ makes a PSG query for some message m, $\mathcal{B}$ returns $(m, \mathcal{O}$**-Sim_*PSG***$(m))$ as the result. The simulated PSG oracle $\mathcal{O}$**-Sim_*P SG*** operates as Fig. 5.

– *PSV queries:* When $\mathcal{A}$ makes a PSV query for some message m, an SDVPS $\delta = (s_1, s_2, T)$ and a warrant m_w, $\mathcal{B}$ returns $\mathcal{O}$**-Sim_*PSV***(m, δ, m_w) as the result. The simulated PSV oracle $\mathcal{O}$**-Sim_*PSV*** operates as Fig. 6.

Analysis of the game For each PCG and PSG query, $\mathcal{B}$ always returns a computationally indistinguishable result. Besides, each simulated random oracle also normally terminates

without collision. Let Fv be the event that $\mathcal{A}$ tries to forge an SDVPS for a message m and then finally outputs a valid SDVPS $\delta = (s_1, s_2, T)$ along with a warrant m_w. By assumption, we know that $\mathcal{A}$ has non-negligible probability ε to break the proposed SDVPS scheme, *i.e.*, $\Pr[\text{Fv}] = \varepsilon$. The probability that $\mathcal{A}$ guesses a correct value without asking $h_2(m, T)$ random oracle is not greater than 2^{-k}. We denote such an event by $(\neg \text{QH}_2)$ and $\Pr[\neg \text{QH}_2] \leq 2^{-k}$. Therefore, we can further express the probability that $\mathcal{A}$ outputs a valid forgery after asking the corresponding h_2 random oracle as

$$\Pr[\text{Fv} \mid \text{QH}_2] \geq (\varepsilon - 2^{-k}).$$

$\mathcal{B}$ again runs $\mathcal{A}$ on input $\{p, q, g, y_v', y_p', y_o\}$ and the same random tape. Since $\mathcal{A}$ is provided with the same sequence of random bits, we know that the ith query he will ask is always the same as the one during the first simulation. For all the oracle queries before $h_2(m, T)$, $\mathcal{B}$ returns identical results as those in the first time. When $\mathcal{A}$ asks $h_2(m, T)$, $\mathcal{B}$ directly gives a new $v_2^* \in_R Z_q$ instead of v_2. Meanwhile, $\mathcal{A}$ is then provided with another different random tape which is also composed of a long sequence of random bits. By the "Forking Lemma", if $\mathcal{A}$ eventually outputs another valid SDVPS $\delta^* = (s_1, s_2^*, T)$ with $h_2(m, T) \neq h_2'(m, T)$, $\mathcal{B}$ would have a chance to solve the DLP by computing

$$x_p = ((s_2 - s_2^*) + \sigma(v_2 - v_2^*))/(v_2^* - v_2). \tag{10}$$

To evaluate $\mathcal{B}$'s success probability, we use the "Splitting lemma" [26] as follows:

Let X and Y be the sets of possible sequences of random bits and random function values supplied to $\mathcal{A}$ before and after the $h_2(m, T)$ query is made by $\mathcal{A}$, respectively. It follows that on inputting a random value $(e \| f)$ for any $e \in X$ and $f \in Y$, $\mathcal{A}$ outputs a valid forgery with the probability of ε, *i.e.*, $\Pr_{e \in X, f \in Y}[\text{Fv}] = \varepsilon$. According to the "Splitting lemma", there is a subset $D \in X$ such that

(a) $\Pr[e \in D] = |D| \cdot |X|^{-1} \geq 2^{-1}\varepsilon$,
(b) $\forall e \in D$, $\Pr_{f \in Y}[\text{Fv}] \geq 2^{-1}\varepsilon$.

From the above definition, we know that if $n \in D$ is the supplied sequence of random bits and random function values given to $\mathcal{A}$ before the $h_2(m, T)$ query is made, then for any sequence of random bits and random function values $f' \in Y$ after the query, $\mathcal{A}$ outputs a valid forgery with the probability of at least $(2^{-1}\varepsilon)^2 = 4^{-1}\varepsilon^2$, *i.e.*,

$$\Pr_{n \in D, f' \in Y}[\text{Fv}] \geq 4^{-1}\varepsilon^2.$$

Since the probability that $\mathcal{A}$ outputs another SDVPS $\delta^* = (s_1, s_2^*, T)$ with $h_2(m, T) \neq h_2'(m, T)$ is $q_{h_2}^{-1}$, we can express the probability that $\mathcal{B}$ solves the DLP with Eq. (10) in the second simulation as

$$(\varepsilon - 2^{-k})(4^{-1}(\varepsilon - 2^{-k})^2)(q_{h_2}^{-1})$$
$$= 4^{-1}(\varepsilon - 2^{-k})^3(q_{h_2}^{-1}).$$

Moreover, the computational steps required for $\mathcal{B}$ during the entire simulation are

$$t' \approx t + t_\lambda(4q_{PCG} + 6q_{PSG} + 6q_{PSV})$$

where t_λ is the time for performing a modular exponentiation over a finite field. ❑

Theorem 2 The proposed SDVPS *sc*heme satisfies the security requirement of non-transferability.

Proof: To generate an SDVPS δ^* intended for himself, U_v first chooses a proper warrant m_w^* and T^*, $s_2^* \in_R Z_q$ to compute

$$R_1^* = y_v^{s_2^*} \bmod p, \tag{11}$$

$$s_1^* = h_3((R_1^* \cdot (T^* y_p y_o^{-h_1(m_w^*, T^*)})^{x_v h_2(m, T^*)}) \bmod q). \tag{12}$$

Here, $\delta^* = (s_1^*, s_2^*, T^*)$ is a valid SDVPS for m. To be precise, the probability that the computed $\delta^* = (s_1^*, s_2^*, T^*)$ and the received $\delta = (s_1, s_2, T)$ are identical is at most 2^{-3k}, *i.e.*, $\Pr[\delta^* = \delta] \le 2^{-3k}$. ❑

Theorem 3 The proposed SDVPS *s*cheme satisfies the security requirement of strong privacy of signer's identity even under the key-compromise attack.

Proof: On the basis of Proxy-Signature-Verification (PSV) algorithm in our proposed scheme, Eq. (8) can be further expressed as

$$\begin{aligned} R_2 &= (T y_p y_o^{-h_1(m_w, T)})^{x_v h_2(m,T)} \bmod p \\ &= (y_v^{d - x_o h_1(m_w, T) + x_p})^{h_2(m,T)} \bmod p. \end{aligned} \tag{8*}$$

It is obvious that even if the proxy signer's private key x_p is compromised, any malicious adversary still needs both the knowledge of secret integer d and the original signer's private key x_o to perform Eq. (8*). Hence, the strong privacy of signer's identity is fulfilled in the proposed scheme even under the key-compromise attack. ❑

4.3 Comparisons

For facilitating the following comparisons, we first define several used notations:

$|x|$: the bit-length of an integer x
T_h: the time for performing a one-way hash function h
T_m: the time for performing a modular multiplication computation
T_e: the time for performing a modular exponentiation computation
T_i: the time for performing a modular inverse computation

The time for performing the modular addition computation is ignored because it is negligible as compared to the above. We compare the proposed scheme with several previously proposed ones including Jakobsson *et al.*'s (JSI for short) [10], Saeednia *et al.*'s (SKM for short) [14], Huang *et al.*'s (HSM for short) [15], Wang's (Wang for short) [24] and Dai *et al.*'s (DYD for short) [23]. Detailed comparisons in terms of the computational and the

communicational efficiency are demonstrated as Tables 2 and 3, respectively. Note that JSI, SKM and HSM schemes can not offer the function of proxy delegation.

Table 2. Comparisons of computational costs among the proposed and other schemes.

Sch. / Item	JSI	SKM	HSM	Wang	DYD	Ours
Type	Probabilistic	Probabilistic	Deterministic	Probabilistic	Probabilistic	Probabilistic
PCG	$3T_e + 2T_m$ *	$3T_e + 2T_m$ *	$3T_e + 2T_m$ *	$4T_e + 4T_m + 2T_h$	$3T_e + 2T_m + 2T_h$	$3T_e + 2T_m + 2T_h$
PSG	$5T_e + 2T_m + T_h$	$T_e + 2T_m + T_h + T_i$	$T_e + T_h$	$T_e + 2T_m + T_h + T_i$	$3T_e + T_m + T_h$	$T_e + T_m + 2T_h$
PSV	$6T_e + 3T_m + T_h$	$3T_e + 2T_m + T_h$	$T_e + T_h$	$4T_e + 4T_m + 2T_h$	$4T_e + 4T_m + 2T_h$	$3T_e + 4T_m + 3T_h$
Total	$11T_e + 5T_m + 2T_h$	$7T_e + 6T_m + 2T_h + T_i$	$5T_e + 2T_m + 2T_h$	$9T_e + 10T_m + 5T_h + T_i$	$10T_e + 7T_m + 5T_h$	$7T_e + 7T_m + 7T_h$

* We adopt Mambo *et al.*'s scheme [5] to generate the proxy information for the evaluated schemes.

Table 3. Comparisons of communicational costs among the proposed and other schemes.

Sch. / Item	JSI	SKM	HSM	Wang	DYD	Ours
Length	$4\|p\| + 4\|q\|$	$\|p\| + 4\|q\|$	$\|p\| + 2\|q\|$	$\|p\| + 3\|q\|$	$3\|p\| + \|q\|$	$3\|q\|$
Bits*	≈ 2688	≈ 1152	≈ 832	≈ 992	≈ 1696	≈ 480

* Without loss of generality, let $|p| \approx 512$ bits and $|q| \approx 160$ bits. To obtain a fair comparison result, the communicational costs for the warrant m_w in Wang's and our schemes are ignored. Note that the communication costs for proxy information in JSI, SKM and HSM are evaluated by adopting Mambo *et al.*'s scheme [5].

In Table 2, although Huang *et al.*'s scheme has the lowest computational costs among all compared ones, their mechanism is not probabilistic, *i.e.*, a signer will always generate the unique signature for an identical message. Moreover, we also found out that their scheme cannot achieve the strong privacy of signer's identity when a signer's private key is accidentally compromised. Overall, among previous SDVPS schemes we conclude that the proposed one has not only shorter signature length, but also lower computational costs.

5. CONCLUSIONS

In this paper, we have proposed an efficient SDVPS scheme for electronic commerce. The proposed scheme preserves the merits of SDVS schemes and proxy signature schemes. The generated SDVPS can only be verified by a designated verifier for guaranteeing confidentiality. Meanwhile, a designated verifier can not transfer the proxy signature to convince any third party of the proxy signer's identity based on the transcript simulation property. Compared with related works (including previous SDVS and SDVPS schemes), our proposed scheme has not only shorter signature length, but also lower computational costs. That is to say, our proposed SDVPS scheme benefits the practical implementation. Besides, we also proved that the proposed scheme achieves the EF-CMA security in the random oracle model.

REFERENCES

1. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, Vol. IT-31, 1985, pp. 469-472.
2. R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, Vol. 21, 1978, pp. 120-126.
3. W. Stallings, *Cryptography and Network Security: Principles and Practices*, 4th ed., Pearson, New Jersey, USA, 2005.
4. B. Meng, S. Wang, and Q. Xiong, "A fair non-repudiation protocol," in *Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design*, 2002, pp. 68-73.
5. M. Mambo, K. Usuda, and E. Okamoto, "Proxy signature for delegating signature operation," in *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, 1996, pp. 48-57.
6. M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," *IEICE Transactions on Fundamentals of Electronic Communications and Computer Science*, Vol. E79-A, 1996, pp. 1338-1354.
7. B. C. Neuman, "Proxy-based authentication and accounting for distributed systems," *Proceedings of the 13th International Conference on Distributed Computing Systems*, 1993, pp. 283-291.
8. V. Varadharajan, P. Allen, and S. Black, "An analysis of the proxy problem in distributed system," in *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy*, 1991, pp. 255-277.
9. S. Kim, S. Park, and D. Won, "Proxy signatures, revisited," in *Proceedings of International Conference on Information and Communications Security*, 1997, pp. 223-232.
10. M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," *Advances in Cryptology – EUROCRYPT*, 1996, pp. 143-154.
11. I. Ray and N. Narasimhamurthi, "An anonymous electronic voting protocol for voting over the Internet," in *Proceedings of the 3rd International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems*, 2001, pp. 188-190.
12. B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," *Advances in Cryptology – CRYPTO*, 1999, pp. 148-164.
13. G. Wang, "An attack on not-interactive designated verifier proofs for undeniable signatures," Cryptology ePrint Archive, Report 2003/243, 2003, http://eprint.iacr.org/2003/243.
14. S. Saeednia, S. Kremer, and O. Markowitch, "An efficient strong designated verifier signature scheme," in *Proceedings of the 6th International Conference on Information Security and Cryptology*, 2003, pp. 40-54.
15. X. Huang, W. Susilo, Y. Mu, and F. Zhang, "Short designated verifier signature scheme and its identity-based variant," *International Journal of Network Security*, Vol. 6, 2008, pp. 82-93.
16. B. Kang, C. Boyd, and E. Dawson, "A novel identity-based strong designated verifier signature scheme," *The Journal of Systems and Software*, Vol. 82, 2009, pp. 270-273.
17. K. Kumar, G. Shailaja, and A. Saxena, "Identity based strong designated verifier sig-

nature scheme," Cryptology ePrint Archive, Report 2006/134, 2006, http://eprint.iacr.org/2006/134.
18. W. Susilo, F. Zhang, and Y. Mu, "Identity-based strong designated verifier signature schemes," *Information Security and Privacy*, Vol. 3108, 2004, pp. 167-170.
19. J. Zhang and J. Mao, "A novel ID-based designated verifier signature scheme," *Information Sciences*, Vol. 178, 2008, pp. 766-773.
20. S. S. M. Chow, "Multi-designated verifiers signatures revisited," *International Journal of Network Security*, Vol. 7, 2008, pp. 348-357.
21. X. Chen, G. Chen, F. Zhang, B. Wei, and Y. Mu, "Identity-based universal designated verifier signature proof system," *International Journal of Network Security*, Vol. 8, 2009, pp. 52-58.
22. F. Y. Yang and C. M. Liao, "A provably secure and efficient strong designated verifier signature scheme," *International Journal of Network Security*, Vol. 10, 2010, pp. 220-224.
23. J. Z. Dai, X. H. Yang, and J. X. Dong, "Designated-receiver proxy signature scheme for electronic commerce," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1, 2003, pp. 384-389.
24. G. Wang, "Designated-verifier proxy signature schemes," *Security and Privacy in the Age of Ubiquitous Computing*, Vol. 181, 2005, pp. 409-423.
25. H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications*, Springer, Berlin, 2002.
26. D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, Vol. 13, 2000, pp. 361-369.

Han-Yu Lin (林韓禹) received his B.A. degree in Economics from the Fu Jen Catholic University, Taiwan in 2001, and his M.S. degree in Information Management from the Huafan University, Taiwan in 2003. Now he is a Ph.D. candidate in the Department of Computer Science of National Chiao Tung University, Taiwan. His research interests include cryptology and network security.

Tzong-Sun Wu (吳宗杉) received his B.S. degree in Electrical Engineering from the National Taiwan University, Taiwan in 1990, and his Ph.D. in Information Management from the National Taiwan University of Science and Technology, Taiwan in 1998. From August 1998 to July 2002, he has been an Assistant Professor in the Department of Information Management of Huafan University. From August 2001 to January 2007, he has been an Associate Professor in the Department of Informatics of Fo Guang University. He is now with the Department of Computer Science, National Taiwan Ocean University. His research interests include information security, watermarking, digital right management, and e-commerce.

Shih-Kun Huang (黃世昆) is a faculty member in the Department of Computer Science and Information Engineering at National Chiao Tung University in Hsinchu, Taiwan and jointly with the Institute of Information Science, Academia Sinica. His research inter-

ests are in open source software engineering, object-oriented technology and software quality. He received his B.S., M.S. and Ph.D. degrees in Computer Science and Information Engineering from National Chiao Tung University in 1989, 1991 and 1996 respectively.