The Journal of Systems and Software 82 (2009) 974–980

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Efficient self-certified proxy CAE scheme and its variants

Tzong-Sun Wu [a], Han-Yu Lin [b,*]

[a] Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung 202, Taiwan, ROC
[b] Department of Computer Science, National Chiao Tung University, No. 1001, Dasyue Road, Hsinchu City 300, Taiwan, ROC

ARTICLE INFO

Article history:
Received 26 May 2008
Received in revised form 30 September 2008
Accepted 19 December 2008
Available online 6 January 2009

Keywords:
Self-certified
Proxy signature
Authenticated encryption
Convertible

ABSTRACT

Elaborating on the merits of proxy signature schemes and convertible authenticated encryption (CAE) schemes, we adopt self-certified public key systems to construct efficient proxy CAE schemes enabling an authorized proxy signer to generate an authenticated ciphertext on behalf of the original signer. To satisfy the requirement of confidentiality, only the designated recipient is capable of decrypting the ciphertext and verifying the proxy signature. A significant advantage of the proposed schemes is that the proxy signature conversion process takes no extra cost, i.e., when the case of a later dispute over repudiation occurs, the designated recipient can easily reveal the ordinary proxy signature for the public arbitration. If needed, the designated recipient can also convince anyone that he is the real recipient. In addition, integrating with self-certified public key systems, our schemes can earn more computational efficiency, since authenticating the public key and verifying the proxy signature can be simultaneously carried out within one-step.

1. Introduction

With the coming of digital world and the rapid development of Internet, the security of on-line transactions has become increasingly important. It is believed that cryptographic techniques can effectively assure the network security. In 1976, Diffie and Hellman (1976) proposed the first public key cryptosystem based on the discrete logarithm problem (DLP) (Delfs and Knebl, 2002; Menezes et al., 1997). Since then, public key systems have been extensively studied and adopted in lots of applications. In the system, every user first picks a self-chosen private key and then computes the corresponding public one which is made public and thus accessible to anyone. A critical issue for using the public key is to first authenticate it so as to prevent the notorious public key substitution attacks. A public key certificate issued by the certificate authority such as X.509 (ISO/IEC 9594-8, 2001) is a commonly applied countermeasure to withstand the attack. However, it also imposes additional burdens on the communication and computation for transmitting and verifying the certificate.

In 1984, Shamir (1984) introduced the identity-based concept and proposed the so-called ID-based public key cryptosystem. In such system, each user's public key is explicitly verified seeing that the identifier of each user is straightly his public key. In relation to each user's private key, the System Authority (SA) is responsible for computing it with a trapdoor one-way hash function. Without the trapdoor information, no one can manipulate the function. It thus can be seen that the ID-based system places much importance on the SA and it has to be trusted, or else it can impersonate any legitimate user by deriving his private key without being detected.

In 1991, Girault (1991) addressed a novel cryptosystem named self-certified public key system to deal with the issue of public key genuineness. For registration, each user first sends his computed partial information of private key to the SA who in turn computes and returns the key pairs of each user. Generally speaking, the self-certified public key system has the following advantages: (i) No public key certificate is needed, since authenticating the public key can be combined with the subsequent cryptographic operations such as the signature verification, (ii) as compared with ID-based one, this system allows each user to freely choose his private key and (iii) the SA cannot easily impersonate any legitimate user without knowing their original partial information of private keys. One can see that the self-certified public key system is a better alternative to implement cryptographic schemes for strengthening the application security and saving more communication overheads and computation efforts.

To provide some specific applications with both authenticity and confidentiality such as the credit card transaction, the contract signing and the delivery of military documents, in 1994, Horster et al. (1994) proposed an authenticated encryption (AE) scheme. Such scheme allows a signer to generate an authenticated ciphertext and only the designated recipient has the ability to decrypt the ciphertext and verify the embedded signature. By using the designated recipient's public key as a vital parameter in the singing process, the signer can prevent anyone other than the person who has the knowledge of the corresponding private key from verifying the

* Corresponding author. Tel.: +886 926 377200.
E-mail address: hanyu.cs94g@nctu.edu.tw (H.-Y. Lin).

0164-1212/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2008.12.040

signature. As compared with the traditional two-step approach, i.e., first sign then encrypt, an AE scheme greatly improves the computational efficiency. Nevertheless, a dishonest signer might repudiate his generated signature and it is even hard for an arbitrator to judge for that only the designated recipient has the ability to verify the signer's signature. To eliminate the weakness, Araki et al. (1999) proposed a convertible limited verifier signature scheme. Yet, their scheme is costly and might be unworkable if the signer is unwilling to cooperate with. Besides, Zhang and Kim (2003) also found out that Araki et al.'s scheme could not withstand a universal forgery attack on an arbitrary chosen-message. In 2002, Wu and Hsu (2002) proposed a convertible authenticated encryption (CAE) scheme. A CAE scheme equips the designated recipient with the ability to solely convert the authenticated ciphertext into an ordinary signature for the public verification in case of a later dispute over repudiation. Later on, Huang and Chang (2003) also introduced another variant of CAE schemes. Unfortunately, Lv et al. (2005) pointed out that both of their schemes have the same security weakness with regard to the message confidentiality.

For facilitating the operation of proxy delegation in an organization, Mambo et al. (1996a,b) proposed the proxy signature schemes enabling an original signer to authorize his signing power to a proxy signer such that the proxy signer can generate a valid proxy signature on behalf of the original one. In general, the proxy delegation can be categorized into the following sorts:

(i) *Full delegation* (Mambo et al., 1996a,b): The proxy signer is given the proxy signing key which is exactly the private key of the original signer.
(ii) *Partial delegation* (Mambo et al., 1996a,b): The original signer further computes a proxy signing key from his own private key. Note that even with this proxy signing key, the proxy signer cannot successfully derive the original signer's private key.
(iii) *Delegation by warrant* (Neuman, 1993; Varadharajan et al., 1991): To delegate his signing power to the proxy signer, the original signer issues a warrant consisting of the information about the period of validity and the identifiers of the original and the proxy signers, etc.
(iv) *Partial delegation with warrant* (Kim et al., 1997): It combines partial delegation and delegation by warrant. A major advantage of the way is that certifying the warrant and verifying the signature can be simultaneously performed within one procedure.

In this paper, we adopt self-certified public key systems to construct novel proxy CAE schemes using partial delegation with warrant. The proposed schemes further provide traditional CAE schemes with the property of proxy delegation. Without any public key certificate, the designated recipient can authenticate the original and the proxy signers' public keys and verify the proxy signature within one single logical step simultaneously, which benefits to reducing the computational cost. Moreover, there is not any extra communication or computation overhead for the designated recipient to obtain the ordinary proxy signature in the proxy signature conversion phase.

The rest of this paper is organized as follows. We present our schemes in Section 2. The comparisons and security analyses will be discussed in Section 3. Finally, a conclusion is given in Section 4.

2. Self-certified proxy CAE schemes

In this section, we present the proposed schemes over a Galois field (Birkhoff and Mac Lane, 1996). Our schemes can be divided into five phases: the user registration, the proxy credential generation, the proxy signature generation and verification, the proxy signature conversion and the recipient proof phases. In the user registration phase, each user first makes a registration to the SA for joining the system. Then the original signer can delegate his signing power to the proxy signer in the proxy credential generation phase. After that, in the proxy signature generation and verification phase, the proxy signer generates an authenticated ciphertext such that only the designated recipient can decrypt the ciphertext and verify the proxy signature. When the case of a later repudiation dispute occurs, the designated recipient reveals the converted proxy signature in the proxy signature conversion phase. If needed, the designated recipient can also prove himself as the real recipient in the recipient proof phase. Fig. 1 depicts the involved parties and the communication flow of the proposed schemes.

Initially, the system determines the following public information:

- p, q: two large primes satisfying that $q|(p-1)$;
- g: a generator of order q over the finite field GF(p);
- $h(\cdot)$: a secure one-way hash function which accepts input of any length and generates a fixed length output;
- γ: the SA's private key $\gamma \in Z_q^*$;
- β: the SA's public key computed as

$$\beta = g^{\gamma} \bmod p. \tag{1}$$

All the above parameters are made public except for the SA's private key γ. Details of each phase are described below:

The user registration phase

To join the system, each user U_i associated with the identifier ID_i has to perform the following interactive steps with the SA to obtain his key pair:

Step 1 U_i first chooses an integer $t_i \in Z_q^*$ to compute

$$v_i = g^{h(t_i, ID_i)} \bmod p, \tag{2}$$

and then delivers (v_i, ID_i) to the SA.

Step 2 Upon receiving (v_i, ID_i), the SA chooses $z_i \in Z_q^*$ to compute

$$y_i = v_i h(ID_i)^{-1} g^{z_i} \bmod p, \tag{3}$$
$$w_i = z_i + h(y_i, ID_i)\gamma \bmod q, \tag{4}$$

and sends (y_i, w_i) back to U_i.

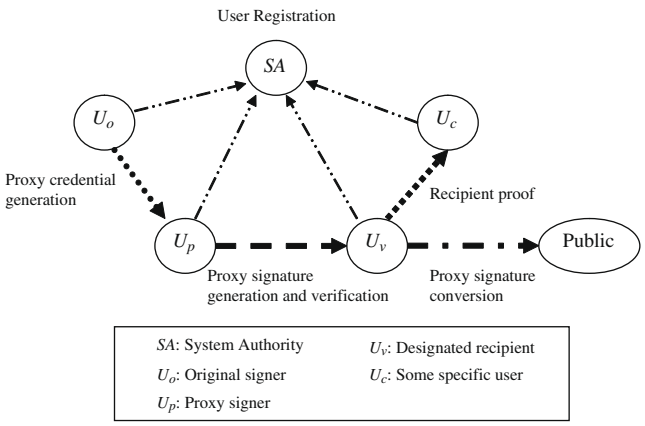

Fig. 1. Illustration of the proposed proxy CAE schemes.

Step 3 U_i computes his private key x_i as

$$x_i = w_i + h(t_i, ID_i) \bmod q, \tag{5}$$

and then ensures its validity by checking

$$\beta^{h(y_i, ID_i)} h(ID_i) y_i \stackrel{?}{=} g^{x_i} (\bmod\ p). \tag{6}$$

If it holds, U_i accepts (x_i, y_i) as his private–public key pair. The correctness of Eq. (6) can be easily confirmed as Theorem 1, which also validates the authenticity of y_i with respect to x_i.

Theorem 1. *A valid key pair (x_i, y_i) can pass the test of Eq. (6).*

Proof. From the left-hand side of Eq. (6), we have

$$\begin{aligned}
\beta^{h(y_i, ID_i)} h(ID_i) y_i &= \beta^{h(y_i, ID_i)} v_i g^{z_i} \quad \text{(by Eq.(3))} \\
&= v_i g^{z_i + h(y_i, ID_i)\gamma} \quad \text{(by Eq.(1))} \\
&= g^{h(t_i, ID_i)} g^{z_i + h(y_i, ID_i)\gamma} \quad \text{(by Eq.(2))} \\
&= g^{h(t_i, ID_i) + w_i} \quad \text{(by Eq.(4))} \\
&= g^{x_i} (\bmod\ P) \quad \text{(by Eq.(5))}
\end{aligned}$$

which equals to the right-hand side of Eq. (6). □

***The proxy credential generation phase*:** Let U_o be the original user delegating his signing power to the proxy signer U_p. U_o distributes the proxy credential to U_p with the following steps:

Step 1 U_o first chooses an integer $t \in_R Z_q^*$ to compute

$$T = g^t \bmod p, \tag{7}$$

$$\sigma = x_o + t(h(m_w, T)) \bmod q, \tag{8}$$

and then sends (σ, m_w, T) to U_p where m_w is the warrant consisting of the identifiers of the original and proxy signers, the delegation duration and so on.

Step 2 Upon receiving (σ, m_w, T), U_p verifies

$$g^{\sigma} = \beta^{h(y_o, ID_o)} h(ID_o) y_o T^{h(m_w, T)} (\bmod\ p) \tag{9}$$

If it holds, U_p proceeds to the next step; else, (σ, m_w, T) is requested to be sent again.

Theorem 2. *The verification of Eq. (9) works correctly.*

Proof. By raising both sides of Eq. (8) to exponent with base g, we have

$$\begin{aligned}
g^{\sigma} &= g^{x_o + t h(m_w, T)} \\
&= \beta^{h(y_o, ID_o)} h(ID_o) y_o g^{t h(m_w, T)} \quad \text{(by Eq.(6))} \\
&= \beta^{h(y_o, ID_o)} h(ID_o) y_o T^{h(m_w, T)} (\bmod\ p) \quad \text{(by Eq.(7))}
\end{aligned}$$

which implies Eq. (9). □

***The proxy signature generation and verification phase*:** For signing the message m on behalf of the original signer U_o, U_p randomly chooses an integer $k \in Z_q^*$ to compute

$$C = (\beta^{h(y_v, ID_v)} h(ID_v) y_v)^k \bmod p, \tag{10}$$

$$r_1 = m(h(C))^{-1} \bmod p, \tag{11}$$

$$r_2 = h(m, h(g^k \bmod p), C) \bmod q, \tag{12}$$

U_p then compute s as Eq. (13.*) in Table 1, where '*' represents one letter of 'a' to 'f'. Each equation is a secure combination of five parameters k, x_p, σ, r_2 and T. Here, (m_w, r_1, r_2, s, T) is the proxy signature which is sent to the designated recipient U_v. Upon receiving it, U_v first computes K and C from Eq. (14.*) of Table 2 and Eq. (15), respectively. Note that the computation of K depends on the generation of s, e.g., generating s with Eq. (13.a) implies deriving K with Eq. (14. a).

Table 1
Equations to generate signature s.

I	$s = k(x_p + \sigma) h(r_2, T) \bmod q$	(13.a)
II	$s = k(1 + (x_p + \sigma) h(r_2, T))^{-1} \bmod q$	(13.b)
III	$s = k(x_p + \sigma)^{-1} h(r_2, T)(x_p + \sigma)^{-1} \bmod q$	(13.c)
IV	$s = k^{-1} h(r_2, T) + k^{-1}(x_p + \sigma) \bmod q$	(13.d)
V	$s = k h(r_2, T)(x_p + \sigma) \bmod q$	(13.e)
VI	$s = k(x_p + \sigma)^{-1} h(r_2, T)(x_p + \sigma)^{-1} \bmod q$	(13.f)

$$C = K^{x_v} \bmod p. \tag{15}$$

U_v then recovers the message m as

$$m = h(C) r_1 \bmod p, \tag{16}$$

and checks the redundancy embedded in m. U_v can further verify the proxy signature (m_w, r_1, r_2, s, T) by checking

$$r_2 = h(m, h(K), C) \bmod q. \tag{17}$$

Take scheme I (with s and K separately computed as Eqs. (13.a) and (14. a)) as an example. We demonstrate that Eqs. (16) and (17) work correctly as the proofs of Theorems 3 and 4, respectively. The correctness of the other schemes can be assured with the similar way.

Theorem 3. *With the proxy signature (m_w, r_1, r_2, s, T), the designated recipient U_v can recover the message m and check its validity with Eq. (16).*

Proof. From the right-hand side of Eq. (16), we have

$$\begin{aligned}
h(C) r_1 &= h(K^{x_v} \bmod p) r_1 \quad \text{(by Eq.(15))} \\
&= h((g^s (\beta^{h(y_o, ID_o) + h(y_p, ID_p)} h(ID_o) h(ID_p) y_o y_p T^{h(m_w, T)})^{h(r_2, T)})^{x_v} \\
&\quad \bmod p) r_1 \quad \text{(by Eq.(14))} \\
&= h((g^{s + (x_p + \sigma) h(r_2, T)})^{x_v} \bmod p) r_1 \quad \text{(by Eq.(9) and (6))} \\
&= h((g^k)^{x_v} \bmod p) r_1 \quad \text{(by Eq.(13))} \\
&= h((\beta^{h(y_v, ID_v)} h(ID_v) y_v)^k \bmod p) r_1 \quad \text{(by Eq.(6))} \\
&= h(C) r_1 \quad \text{(by Eq.(10))} \\
&= m (\bmod\ p) \quad \text{(by Eq.(11))}
\end{aligned}$$

which leads to the left-hand side of Eq. (16). □

Theorem 4. *If the proxy signature (m_w, r_1, r_2, s, T) is correctly generated, it will pass the test of Eq. (17).*

Proof. From the right-hand side of Eq. (17), we have

$$\begin{aligned}
&h(m, h(K), C) \\
&= h(m, h(K), K^{x_v} \bmod p) \quad \text{(by Eq.(15))} \\
&= h(m, h(g^s (\beta^{h(y_o, ID_o) + h(y_p, ID_p)} h(ID_o) h(ID_p) y_o y_p T^{h(m_w, T)})^{h(r_2, T)} \\
&\quad \bmod p), (g^s (\beta^{h(y_o, ID_o) + h(y_p, ID_p)} h(ID_o) h(ID_p) y_o y_p T^{h(m_w, T)})^{h(r_2, T)} \\
&\quad \bmod p)^{x_v} \bmod p) \quad \text{(by Eq.(14))} \\
&= h(m, h(g^{s + (x_p + \sigma) h(r_2, T)} \bmod p), g^{(s + (x_p + \sigma) h(r_2, T)) x_v} \\
&\quad \bmod p) \quad \text{(by Eqs.(9) and (6))} \\
&= h(m, h(g^k \bmod p), g^{k x_v} \bmod p) \quad \text{(by Eq.(13))} \\
&= h(m, h(g^k \bmod p), C) \quad \text{(by Eqs.(10) and (6))} \\
&= r_2 (\bmod\ q) \quad \text{(by Eq.(12))} \quad \text{(by Eq.(1))}
\end{aligned}$$

which leads to the left-hand side of Eq. (17). □

***The proxy signature conversion phase*:** When the case of a later dispute over repudiation occurs, the designated recipient U_v can reveal the converted proxy signature (m_w, r_2, s, C, T) and the origi-

Table 2
Equations to compute K.

I	$K = g^s(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)})^{h(r_2,T)} \bmod p$	(14.a)
II	$K = g^s(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)})^{sh(r_2,T)} \bmod p$	(14.b)
III	$K = (g(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)})^{s})^{h(r_2,T)^{-1}} \bmod p$	(14.c)
IV	$K = (g^{h(r_2,T)}(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)}))^{s^{-1}} \bmod p$	(14.d)
V	$K = (g^s(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)}))^{h(r_2,T)^{-1}} \bmod p$	(14.e)
VI	$K = g^{h(r_2,T)}(\beta^{h(y_o,ID_o)+h(y_p,ID_p)}h(ID_o)h(ID_p)y_o y_p T^{h(m_w,T)})^{s} \bmod p$	(14.f)

nal message m to prove the proxy signer's dishonesty without extra computation efforts or communication overheads. Thus, anyone can verify the converted proxy signature with the assistance of Eqs. (14.*) and (17).

***The recipient proof stage*:** For convincing someone, say, U_c, that he is the real recipient, the recipient U_v can perform the following interactive steps with U_c:

Step 1 U_v sends (m_w, r_2, s, C, T) and m to U_c.
Step 2 U_c first computes K with the corresponding Eq. (14.*) and then checks the signature's validity with Eq. (17). If it holds, U_c proceeds to the next step; otherwise, the protocol is terminated.
Step 3 U_c randomly chooses an integer e to compute $E = K^e \bmod p$ and then transmits E to U_v.
Step 4 Upon receiving E, U_v computes $W = E^{x_v} \bmod p$ and returns it to U_c.
Step 5 U_c computes $W' = C^e \bmod p$ and checks whether $W = W'$. If it holds, U_c is convinced that U_v is the designated recipient.

3. Security analyses

In this section, we first define some security notions with respect to self-certified proxy CAE schemes and then give security proofs, comparisons and the performance evaluation of our proposed schemes.

3.1. Security notions

To facilitate the following proofs, we regenerate algorithms of UR, PCG, PSiG/V, PSC and RP from the five phases of our proposed scheme I. The security notions of message confidentiality and unforgeability with respect to self-certified proxy CAE schemes are defined below.

3.1.1. Message confidentiality

A self-certified proxy CAE scheme can fulfill the security requirement of message confidentiality if the authenticated ciphertext is indistinguishable under chosen ciphertext attacks. We define a security model for indistinguishability of the authenticated ciphertext under chosen ciphertext attacks. In this model, the adversary attempts to decrypt a target ciphertext of the designated recipient.

Definition 1. A self-certified proxy CAE scheme is said to be semantically secure against chosen ciphertext attacks if there exits no polynomial-time adversary with a non-negligible advantage in the following game:

Setup: A challenger $\mathscr{C}$ first generates necessary system parameters and then obtains key pairs of one original signer U_o and one proxy signer U_p by the UR algorithm. System parameters and the public keys of U_o and U_p are given to an adversary $\mathscr{A}$. Upon receiving these parameters, the adversary $\mathscr{A}$ determines one designated recipient U_v^*. The recipient U_v^*'s public key y_v^* can be acquired from the UR algorithm, but the corresponding private key x_v^* is unknown to $\mathscr{A}$.

Phase 1: The adversary $\mathscr{A}$ can issue several kinds of queries adaptively:

- PCG queries: The adversary $\mathscr{A}$ sends a PCG query for U_o and U_p to the challenger $\mathscr{C}$. The adversary will be given the result of PCG(m_w, x_o) by $\mathscr{C}$.
- PSiG/V queries: The adversary $\mathscr{A}$ can query either PSiG or PSiV. In the PSiG queries, the adversary $\mathscr{A}$ produces a message m with respect to U_o, U_p and U_v^* and sends it to the challenger $\mathscr{C}$ which then returns the result of PSiG$(m, m_w, \sigma, x_p, y_v^*)$ to $\mathscr{A}$. In the PSiV queries, the adversary $\mathscr{A}$ produces an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$ and requests the result of PSiV$(\delta, m_w, y_o, y_p, x_v^*)$ with respect to U_o, U_p and U_v^* from the challenger $\mathscr{C}$. If the recovered message is consistent with the redundancy check and its corresponding signature is valid, $\mathscr{C}$ responses the message; Otherwise, the $\perp$ symbol is returned as a result.
- PSC queries: The adversary $\mathscr{A}$ produces an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$ and requests the result of PSC$(\delta, m_w, y_o, y_p, x_v^*)$ with respect to U_o, U_p and U_v^* from the challenger $\mathscr{C}$. If the result (m_w, r_2, s, C, T) is a valid converted signature for the message m with suitable redundancy, $\mathscr{C}$ responses the result; Otherwise, the $\perp$ symbol is returned as a result.
- RP queries: The adversary $\mathscr{A}$ produces an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$ and requests the result of RP$(\delta, m_w, y_o, y_p, x_v^*)$ with respect to U_o and U_p from the challenger $\mathscr{C}$. If U_v^* is the designated recipient, $\mathscr{C}$ responses the symbol 1; Otherwise, the $\perp$ symbol is returned as a result.

Challenge: The adversary $\mathscr{A}$ produces two messages, m_0 and m_1, of the same length. The challenger $\mathscr{C}$ flips a coin $\lambda \leftarrow \{0,1\}$ and generates an authenticated ciphertext $\delta^* = \text{PSiG}(m_\lambda, m_w, \sigma, x_p, y_v^*)$ which is then delivered to $\mathscr{A}$ as a target challenge.

Phase 2: The adversary $\mathscr{A}$ can issue new queries as those in Phase 1, except that the PSiV or PSC query for the target challenge δ^* is prohibited.

Guess: At the end of the game, $\mathscr{A}$ outputs a bit λ'. The adversary $\mathscr{A}$ wins this game if $\lambda' = \lambda$. We define $\mathscr{A}$'s advantage as $Adv(\mathscr{A}) = \Pr[\lambda' = \lambda] - 1/2$.

3.1.2. Unforgeability

A cryptographic scheme satisfies the security requirement of unforgeability if it is secure against chosen-message attacks. We define a model for unforgeability of self-certified proxy CAE scheme against chosen-message attacks. In this model, the adversary attempts to forge a valid signature of one target message.

Definition 2. A self-certified proxy CAE scheme is said to achieve existential unforgeability against chosen-message attacks if there exists no polynomial-time adversary with a non-negligible advantage in the following game:

Setup: A challenger $\mathscr{C}$ first generates necessary system parameters, and then obtains key pairs of an original signer U_o and a proxy signer U_p, and a designated recipient U_v''s key pair (x_v, y_v) by the UR algorithm. The challenger $\mathscr{C}$ then gives the forger $\mathscr{F}$ system parameters, public keys of U_o, U_p and U_v.

Attack: The forger $\mathscr{F}$ issues the same queries as those in Phase 1 of Definition 1.

Forgery: Finally, $\mathscr{F}$ produces an authenticated ciphertext δ^*. The forger $\mathscr{F}$ wins if δ^* can be converted into a valid signature $(m_w, r_2^*, s^*, C^*, T)$ for some message m^* with redundancy by the designated recipient. Note that it is not allowed to issue a PSiG query for m^*.

3.2. Security proof

The mathematical assumption of our schemes is the discrete logarithm problem (DLP) (Delfs and Knebl, 2002; Diffie and Hellman, 1976; Menezes et al., 1997) as well as the security of Nyberg–Rueppel signature schemes (Nyberg and Rueppel, 1993, 1994). For the details of Nyberg–Rueppel signature schemes, interested readers can refer to (Nyberg and Rueppel, 1993, 1994). The definitions of DLP are restated below:

Definition 3 (*discrete logarithm problem; DLP*). Let (p, q) be two large primes satisfying that $q|p-1$ and g a generator of order q over GF(p). The discrete logarithm problem is, given an instance (y,p,q,g) for some $y \in Z_p^*$, to derive $x \in Z_q$ such that $y = g^x \bmod p$. Here, we denote the discrete logarithm $x = \mathrm{Log}_{p,q,g}(y)$.

Definition 4 (*discrete logarithm assumption*). Let $I_k = \{(p,q,g) \in I || p| = k\}$ with $k \in N$, where I is the universe of all instances and $|p|$ represents the bit-length of p. For every probabilistic polynomial-time algorithm $\mathscr{S}$, every positive polynomial $P(\cdot)$ and all sufficiently large k, the algorithm $\mathscr{S}$ can solve the DLP with an advantage at most $\frac{1}{P(k)}$, i.e.,

$$\Pr[\mathscr{S}(y,p,q,g) = \mathrm{Log}_{p,q,g}(y), (p,q,g) \xleftarrow{u} I_k, y \xleftarrow{u} Z_p^*] \leqslant \frac{1}{P(k)}.$$

Note that "$\xleftarrow{u}$" denotes uniformly and independently selected. The probability is taken over the uniformly and independently chosen instance with a given security parameter k and over the random choices of $\mathscr{S}$.

Instead of separate discussions, we only take the scheme I as an instance for the following proofs. One can see that the proposed schemes (I–VI) have their different equations to generate s and K. Interested readers can replace the corresponding equations and follow the construction to prove other schemes with similar ways. Theorems 5 and 6 prove that the proposed scheme achieves the security requirements of confidentiality and unforgeability, respectively.

Theorem 5. *The proposed self-certified proxy CAE scheme is* (t, ε)*-secure against chosen ciphertext attacks if there exists no polynomial-time algorithm* β_1 *that can* (t_1, ε_1)*-break the DLP.*

Proof. Suppose that $\mathscr{A}$ is a (t, ε)-algorithm that breaks the self-certified proxy CAE scheme under the chosen ciphertext attack, where t denotes the running time and ε the probability that $\mathscr{A}$ succeeds. We will show that we can use $\mathscr{A}$ to construct a (t_1, ε_1)-algorithm β_1 that solves the DLP in time t_1 with the probability ε_1. The algorithm β_1 is said to (t_1, ε_1)-break the DLP. Let $(g, g^\alpha \bmod p)$ be a random instance of the DLP. The objective of the algorithm β_1 is to derive α. In this proof, β_1 simulates challenger $\mathscr{C}$ in the game of Definition 1. In the meantime, $\mathscr{A}$ adaptively issues queries as those defined in the game of Definition 1. □

- PCG queries: When $\mathscr{A}$ issues a PCG query for U_o and U_p, the algorithm β_1 first randomly chooses an integers $t \in_R Z_q^*$ to compute $T = g^t \bmod p$ and $\sigma = x_o + t(h(m_w, T)) \bmod q$. (σ, m_w, T) is then returned as the result of PCG query for U_o and U_p.
- PSiG queries: When $\mathscr{A}$ issues a PSiG query on a message m, β_1 first randomly chooses an integer $k \in Z_q^*$ and computes $C = (\beta^{h(y_v^*, ID_v)} h(ID_v) y_v^*)^k \bmod p$. Then, β_1 computes $r_1 = mh(C)^{-1} \bmod p$, $r_2 = h(m, h(g^k \bmod p), C) \bmod q$, and $s = k - (x_p + \sigma)h(r_2, T) \bmod q$. Here, (m_w, r_1, r_2, s, T) is the authenticated ciphertext δ which is returned as the result of the PSiG on the message m.
- PSiV queries: When $\mathscr{A}$ issues a PSiV query on an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$, β_1 first computes $K = g^s (\beta^{h(y_o, ID_o) + h(y_p, ID_p)} h(ID_o) h(ID_p) y_o y_p T^{h(m_w, T)})^{h(r_2, T)} \bmod p$ and $C = K^{x_v^*} \bmod p$. Then, β_1 recovers $m = h(C) r_1 \bmod p$. If the recovered m is consistent with the redundancy check and the equality of $r_2 = h(m, h(K), C) \bmod q$ holds, β_1 outputs m; otherwise, the $\perp$ symbol is returned as a result.
- PSC queries: When $\mathscr{A}$ issues a PSC query on an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$, the algorithm β_1 first computes $K = g^s (\beta^{h(y_o, ID_o) + h(y_p, ID_p)} h(ID_o) h(ID_p) y_o y_p T^{h(m_w, T)})^{h(r_2, T)} \bmod p$ and $C = K^{x_v^*} \bmod p$. Then, β_1 recovers $m = h(C) r_1 \bmod p$. If the result (m_w, r_2, s, C, T) satisfies $r_2 = h(m, h(K), C) \bmod q$, outputs the result; Otherwise, the $\perp$ symbol is returned as result.
- RP queries: When $\mathscr{A}$ issues a RP query on an authenticated ciphertext $\delta = (m_w, r_1, r_2, s, T)$, β_1 first performs the same steps as those in PSC queries, and then chooses an integer e to compute $E = K^e \bmod p$, $W = E^{x_v *} \bmod p$, and $W' = C^e \bmod p$. If $W = W'$, β_1 outputs the symbol 1 as the result. Otherwise, the $\perp$ symbol is returned as result.

Challenge: The adversary $\mathscr{A}$ generates two messages, m_0 and m_1, of the same length. The challenger β_1 flips a coin $\lambda \leftarrow \{0, 1\}$ and computes an authenticated ciphertext $\delta^* = \mathrm{PSiG}(m_\lambda, m_w, \sigma, x_p, y_v^*)$. The algorithm β_1 first randomly chooses an integer $Z \in Z_q^*$ and computes $C^* = (\beta^{h(y_v^*, ID_v)} h(ID_v) y_v^*)^Z \bmod p$. Then, β_1 computes $r_1^* = m_\lambda h(C^*)^{-1} \bmod p$, $r_2^* = h(m_\lambda, h(g^\alpha \bmod p), C^*) \bmod q$, and $s^* = Z - (x_B + \sigma)h(r_2^*, T) \bmod q$. The authenticated ciphertext $\delta^* = (m_w, r_1^*, r_2^*, s^*, T)$ is sent to $\mathscr{A}$ as the target challenge. If $Z = \alpha$, then δ^* is indeed a random PSiG of m_λ. If Z is a random integer and does not equal to α, then r_1^* and s^* are random elements. Therefore, δ^* is independent of λ.

Phase 2: The adversary $\mathscr{A}$ issues new queries as those in Phase 1. It is not allowed to make a PSiG or PSC query for the target challenge δ^*.

Analysis: Consider the case when $Z = \alpha$, the distribution of the adversary $\mathscr{A}$'s view in the simulation is identical to that $\mathscr{A}$ is playing the game with C. Consequently, $\Pr_{\beta_1}[\mathrm{Succ}] = \Pr_{\mathscr{A}}[\mathrm{Succ}] - 1/2$, where $\Pr_{\mathscr{A}}[\mathrm{Succ}]$ stands for the probability that $\mathscr{A}$ succeeds. When Z is uniformly distributed in Z_q^*, the adversary $\mathscr{A}$ has no information about the value of λ and hence the probability of $\lambda' = \lambda$ is at most 1/2. Therefore, we conclude that $\Pr_{\beta_1}[\mathrm{Succ}] = \varepsilon_1 \geqslant \Pr_{\mathscr{A}}[\mathrm{Succ}] - 1/2 = \varepsilon - 1/2$.

Remark 1. Note that the result in Theorem 5 also proved the computational secrecy of the proposed scheme. Any adversary who has the ability to guess the right message candidate for a given ciphertext with the probability greater than 1/2 can win the simulation game and then construct the algorithm β_1 to solve the DLP. In other words, on condition that the discrete logarithm assumption is intractable, we can claim that the proposed scheme is secure against the chosen ciphertext attack.

Theorem 6. *The proposed self-certified proxy CAE scheme is* (t, ε)*-secure against existential forgery under chosen plaintext attacks if there exists no polynomial-time algorithm* β_2 *that can forge the Nyberg–Rueppel signature in time* t_2 *with the probability* ε_2.

Proof. Suppose that $\mathscr{F}$ is a (t, ε)-algorithm that breaks the self-certified proxy CAE scheme under chosen-message attacks in time t with the probability ε. We will construct a (t_2, ε_2)-algorithm β_2 that forges the Nyberg–Rueppel signature in time t_2 with the probability ε_2 from the algorithm $\mathscr{F}$. The objective of the algorithm β_2 is to derive a valid Nyberg–Rueppel signature. In this proof, β_2 simulates $\mathscr{F}$'s challenger in the game of Definition 2 with the target proxy signer U_p's public key y_p^* where $\beta^{h(y_p^*, ID_p)} h(ID_p) y_p^* = g^{x_p^*} \bmod p$. Then, $\mathscr{F}$ adaptively issues the same queries as those defined in the game of Definition 1. □

Forgery: The algorithm $\mathscr{F}$ generates an authenticated ciphertext $\delta^* = (m_w, r_1^*, r_2^*, s^*, T)$ for one target message m^* under the private key of the designated recipient. Note that δ^* is not obtained from a PSiG query of $(m^*, m_w, \sigma, x_p^*, y_v)$.

Analysis: $\mathscr{F}$ outputs an authenticated ciphertext δ^* which can be converted to the message m^* and its corresponding signature $(m_w, r_2^*, s^*, C^*, T)$ with a non-negligible probability. If $(m_w, r_2^*, s^*, C^*, T)$ satisfies the signature verification equation $r_2^* = h(m^*, h(K^*), C^*) \bmod q$ with the probability ε, then (r_2^*, S^*) can be regarded as a valid Nyberg–Rueppel signature of the message m^* with respect to the public key y_p^*. It can be seen that $\Pr_{\beta_2}[\text{Succ}]$ is hence at least $Pr_{\mathscr{F}}[\text{Succ}]$. We conclude that $\Pr_{\beta_2}[\text{Succ}] = \varepsilon_2 \geqslant \varepsilon$.

Remark 2. Theorem 6 uses the notion of reducing the Nyberg–Rueppel signature scheme to the proposed scheme. If an adversary has the ability to forge a valid signature for a target plaintext of our scheme, he can follow the similar construction to forge a valid Nyberg–Rueppel signature. Hence, based on the security of Nyberg–Rueppel signature scheme, we can claim that the proposed scheme is secure against existential forgery under chosen plaintext attacks.

3.3. Comparisons and the performance evaluation

Integrating with self-certified public systems, the proposed proxy CAE schemes also inherit the merit that authenticating the public key and verifying the signature can be simultaneously combined into one logical procedure, which benefits to reduce the cost for transmitting and verifying the public key certificate. The message recovery property allows the designated recipient to recover the original message from the received authenticated ciphertext, which also helps more bandwidth saving. Besides, the proxy signature conversion is rather simple and takes no extra computation efforts or communication overheads, because the converted proxy signature will be derived during the proxy signature verification process. In fact, the proposed schemes are the first models combing the CAE scheme with self-certified public key systems and the concept of proxy delegation. That is, the proposed schemes are totally different from previously proposed ones. Consequently, we only make the functionality comparison with some previous CAE schemes including Araki et al.'s scheme (Araki et al., 1999) (AUI for short), the Wu–Hsu scheme (Wu and Hsu, 2002) (WH for short), the Huang–Chang scheme (Huang and Chang, 2003) (HC for short) and Lv et al.'s scheme (Lv et al., 2005) (LWK for short). Then we will evaluate the performance of our proposed schemes. Detailed comparisons are listed as Table 3. From this table, one can observe that only Lv et al.'s scheme and the proposed ones achieve the requirement of computational secrecy. Although Lv et al.'s scheme supplies almost as many functionalities as the proposed ones, their scheme cannot deal with the issue of proxy delegation which is considered to be an essential operation in an organization. Therefore, we conclude that the proposed schemes provide better functionalities.

To facilitate the reader with the performance evaluation, we define the following notations:

Table 3
Functionality comparisons of the proposed and other schemes.

Functionality	Scheme				
	AUI	WH	HC	LWK	WL*
Message recovery	O	O	O	O	O
Without public key certificate	×	×	×	O	O
Proxy delegation	×	×	×	×	O
Signature conversion	O	O	O	O	O
No conversion cost	×	O	O	O	O
Computational secrecy	×	×	×	O	O
Recipient proof	×	×	×	O	O

Remark: * WL stands for the proposed schemes.

Table 4
Communication overheads of the proposed schemes.

Item	Authenticated ciphertext	Converted proxy signature
Parameters	(m_w, r_1, r_2, s, T)	(m_w, r_2, s, C, T)
Bit-length	$2\|p\| + 3\|q\|$	$2\|p\| + 3\|q\|$

Table 5
Computation complexities of the proposed schemes.

Phase			Computation complexities
User registration	I–VI	U_i	$4T_h + 2T_m + 3T_e$
		SA	$2T_h + 3T_m + T_e + T_i$
Proxy credential generation	I–VI	U_o	$T_h + T_m + T_e$
		U_p	$3T_h + 3T_m + 3T_e$
Proxy signature generation and verification	I	U_p	$6T_h + 4T_m + 3T_e + T_i$
		U_v	$9T_h + 7T_m + 5T_e$
	II	U_p	$6T_h + 5T_m + 3T_e + 2T_i$
		U_v	$9T_h + 8T_m + 5T_e$
	III	U_p	$6T_h + 5T_m + 3T_e + 2T_i$
		U_v	$9T_h + 7T_m + 5T_e + T_i$
	IV	U_p	$6T_h + 5T_m + 3T_e + 2T_i$
		U_v	$9T_h + 7T_m + 5T_e + T_i$
	V	U_p	$6T_h + 4T_m + 3T_e + T_i$
		U_v	$9T_h + 7T_m + 5T_e + T_i$
	VI	U_p	$6T_h + 5T_m + 3T_e + 2T_i$
		U_v	$9T_h + 7T_m + 5T_e$
Proxy signature conversion	I–VI	U_v	0
Recipient proof	I	U_v	T_e
		U_c	$6T_h + 6T_m + 6T_e$
	II	U_v	T_e
		U_c	$6T_h + 7T_m + 6T_e$
	III	U_v	T_e
		U_c	$6T_h + 6T_m + 6T_e + T_i$
	IV	U_v	T_e
		U_c	$6T_h + 6T_m + 6T_e + T_i$
	V	U_v	T_e
		U_c	$6T_h + 6T_m + 6T_e + T_i$
	VI	U_v	T_e
		U_c	$6T_h + 6T_m + 6T_e$

$|x|$: 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_i: the time for performing a modular inverse computation;
T_e: the time for performing a modular exponentiation computation.

Note that the time for performing the modular addition/subtraction is ignored because it is negligible as compared to those of performing other computations. The detailed evaluation with respect to communication overheads and computation complexities of the proposed schemes is shown as Tables 4 and 5, respectively. From Table 4, one can find that the bit-length of authenticated ciphertext is the same as that of converted proxy signature, i.e., $2|p| + 3|q|$. As to the computational complexities of the proposed schemes, it can be seen from Table 5 that scheme I has the best performance in terms of the total computation of entire protocol. To be precise, when we only consider the complexities of proxy signer, schemes I and V outperform the others. On the contrary, if the complexities of designated recipient is concerned the most, schemes I and IV are better alternatives.

4. Conclusions

This paper has presented efficient self-certified proxy convertible authenticated encryption (CAE) schemes using partial delegation with warrant. The proposed schemes allow the proxy signer to generate a valid authenticated ciphertext on behalf of the original

signer such that only the designated recipient is capable of recovering the original message and verifying its corresponding proxy signature. Preserving the merit of self-certified public key systems, the proposed schemes do not have to transmit and verify the public key certificate in advance, because authenticating the public key and verifying the proxy signature can be combined within one single procedure. Besides, the designated recipient is equipped with the ability to prove himself as the real recipient and can easily reveal the ordinary proxy signature to anyone to show the signer's dishonesty when a later dispute over repudiation occurs. We also demonstrate that the proposed schemes are computationally secure on condition that the discrete logarithm assumption is intractable.

References

Araki, S., Uehara, S., Imamura, K., 1999. The limited verifier signature and its application. IEICE Transactions on Fundamentals E82-A (1), 63–68.

Birkhoff, G., Mac Lane, S., 1996. A Survey of Modern Algebra, fifth ed. Macmillan.

Delfs, H., Knebl, H., 2002. Introduction to Cryptography: Principles and Applications. Springer.

Diffie, W., Hellman, M., 1976. New directions in cryptography. IEEE Transactions on Information Theory IT-22 (6), 644–654.

Girault, M., 1991. Self-certified public keys. Advances in Cryptology EUROCRYPT'91. Springer, Berlin, pp. 491–497.

Horster, P., Michel, M., Peterson, H., 1994. Authenticated encryption schemes with low communication costs. Electronics Letters 30 (15), 1212–1213.

Huang, H., Chang, C., 2003. An efficient convertible authenticated encryption scheme and its variant. In: Proceedings of International Conference on Information and Communications Security (ICICS'03). Springer-Verlag, pp. 382–392.

ISO/IEC 9594-8, 2001. Information technology open systems interconnection the directory: public-key and attribute certificate frameworks. International Organization for Standardization.

Kim, S., Park, S., Won, D., 1997. Proxy signatures, revisited. In: Proceedings of International Conference on Information and Communications Security (ICICS'97). Springer-Verlag, pp. 223–232.

Lv, J., Wang, X., Kim, K., 2005. Practical convertible authenticated encryption schemes using self-certified public keys. Applied Mathematics and Computation 169 (2), 1285–1297.

Mambo, M., Usuda, K., Okamoto, E., 1996a, Proxy signature for delegating signature operation. In: Proceedings of the Third ACM Conference on Computer and Communications Security. ACM press, pp. 48–57.

Mambo, M., Usuda, K., Okamoto, E., 1996b. Proxy signatures: delegation of the power to sign messages. IEICE Transactions on Fundamentals E79-A (9), 1338–1354.

Menezes, A., Oorschot, P., Vanstone, S., 1997. Handbook of Applied Cryptography. CRC Press, Inc.

Neuman, B.C., 1993. Proxy-based authentication and accounting for distributed systems. In: Proceedings of the 13th International Conference on Distributed Computing Systems, pp. 283–291.

Nyberg, K., Rueppel, R.A., 1993. A new signature scheme based on the DSA giving message recovery. In: Proceedings of the First ACM Conference on Computer and Communication Security. ACM Press, pp. 58–61.

Nyberg, K., Rueppel, R.A., 1994. Message recovery for signature schemes based on the discrete logarithm problem. Advances in Cryptology – EUROCRYPT'94. Springer, pp. 182–193.

Shamir, A., 1984. Identity-based cryptosystems and signature schemes. Advances in Cryptology – CRYPTO'84, Springer, pp. 47–53.

Varadharajan, V., Allen, P., Black, S., 1991. An analysis of the proxy problem in distributed system. In: Proceedings of 1991 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 255–277.

Wu, T.S., Hsu, C.L., 2002. Convertible authenticated encryption scheme. The Journal of Systems and Software 62 (3), 205–209.

Zhang, F., Kim, K., 2003. A universal forgery on Araki et al.'s convertible limited verifier signature scheme. IEICE Transactions on Fundamentals E86-A (2), 515–516.

Tzong-Sun Wu received his B.S. degree in electrical engineering from the National Taiwan University in 1990 and his Ph.D. in information management from the National Taiwan University of Science and Technology in 1998. He was an Assistant Professor at the Huafan University from August 1998 to July 2001. From August 2001 to July 2002 and from August 2002 to January 2007, he was an Associate Professor at the Huafan University and the Fo Guang University, respectively. He jointed the faculty of the Department of Computer Science and Engineering at the National Taiwan Ocean University in February 2007. His research interests include information security, cryptographic protocol design, digital watermarking, and digital right management.

Han-Yu Lin received his B.A. degree in economics from the Fu-Jen University in 2001 and his M.S. degree in information management from the Huafan University in 2003. Now he is a doctoral student in the Department of Computer Science at the National Chiao Tung University. His research interests include cryptography and network security.