A Generic Conversion from Proxy Signatures to Certificate-based Signatures

Rufen Huang[1,2], Zhenjie Huang[1,3], Qunshan Chen[1,2]

[1] School of Computer Science, Minnan Normal University, China
[2] Key Laboratory of Data Science and Intelligence Application, Fujian Province University, China
[3] Fujian Key Laboratory of Granular Computing, China
hrf@mnnu.edu.cn, zjhuang@mnnu.edu.cn, qschen@mnnu.edu.cn

Abstract

The proxy signature (*PS*) and the certificate-based signature (*CBS*) are both popular cryptographic protocol. The former is a special signature which allows an entity to delegate his signing rights to another, while the later is another attractive cryptography primitive whose original motivation is to simplify certificate's management and to eliminate key escrow problem. However, there is a drawback in the existing security model of *PS*, and there are something in common between the *CBS* and the *PS*. In the paper, we first analyze the drawback of the existing security model of *PS*. Secondly, we propose an improved security model for *PS* which is stronger than the existing one to overcome its drawback, new model allows an adversary of *PS* to issue both queries for different proxy signers but the same original signer. Thirdly, we proposed a new paradigm *PS*-2-*CBS* which is a generic conversion from an existing secure *PS* to a secure *CBS* after analyzed the relationship between the *CBS*s and the *PS*s. and prove that our *PS*-2-*CBS* is secure if the underlying *PS* is secure under improved security model of *PS*. Finally, an example of *PS*-2-*CBS* is gave.

Keywords: Certificate-based signature, Conversion, Delegation, Proxy signature, Security model

1 Introduction

Proxy signature (*PS*) is a special signature which allows an entity to delegate its signing rights to another, and it was invented y Mambo et al. [1]. In a *PS*, there are two entities involved, including an original signer and a proxy signer. A *PS* protocol allows an original signer to delegate its signing power to a proxy signer, who can sign messages on behalf of the original signer. The *PS* was found a lot of practical applications, particularly in mobile communications [2], electronic commerce [2] and distributed computing [3] etc., where delegation of signing rights is very common. According to the types of delegation, the proxy signature can be classified into full delegation, partial delegation, delegation by warrant, and partial delegation with warrant [4]. A number of proxy signature schemes have been introduced, such as partial delegation [1], delegation by warrant ([3] and [5]), and partial delegation with warrant [4]. Among them, the full delegation systems are the least secure and impractical in practice, and the delegation by warrant systems are more practical, and are used more generally. The research of *PS* have aroused great interest of scholars, various *PS* schemes have been proposed, such as *PS* with revocable anonymity ([6] and [7]), quantum *PS* ([8] and [9]), transitive [10], ID-based *PS* [11], lattice-Based *PS* [12], and attribute-Based *PS* [13].

The certificate-based cryptography (*CBC*) was first proposed by Gentry [14] in Eurocrypt 2003, whose original motivation is to simplify certificate management procedures, The certificate-based signature (*CBS*) was introduced by Kang et al. [15] to extending the idea of Gentry's *CBC*. The *CBS* simplified use and management of certificates in the conventional PKI-based signature system and to overcome key escrow problem in identity-based signature system [16]. There are a Certificate Authority (CA) and a signer in a *CBS* scheme. The signer generates himself key pair and requests a certificate from the CA, while the certificate in a *CBS* as a part of the signing key, and the public key be included in the certificate which corresponds to signer. In this way, there isn't to check the existence of certificate. Since Kang et al.'s [15] first *CBS* scheme, a number of definitions, security models and schemes of *CBS* are presented continually, such as Li et al.'s security model and efficient construction of *CBS* [17], Au et al.'s certificate-based (linkable) ring signature scheme [18], Kumar et al.'s proxy blind *CBS* scheme [19], Li et al. *CBS* scheme without pairings [20]. In addition, there are some extensions of the basic *CBS* schemes, such as Huang et al.'s blind scheme [21], and Ma et al.'s aggregation scheme [22].

However, little work has been conducted to deal with the conversion between the *CBS* and the *PS*. In

*Corresponding Author: Rufen Huang; E-mail: hrf@mnnu.edu.cn
DOI: 10.3966/160792642021012201020

2014, Huang et al. [23] proposed a generic construction from certificate-based Signature to proxy signature. There are still no paper about the conversion from *PS* to *CBS*. The *CBS* simplifies the use and management of certificates, and overcomes key escrow problem, and it has drawn much attention due to its unique advantages and has gained many achievements over the years. In the paper, we propose a generic construction *PS*-2-*CBS* from an existing secure *PS* to a *CBS*.

The contributions of the paper are summarized as follows. First, we analyze the definitions and security model for *CBS* and *PS* delegated by warrant, and illustrate that the existing security model of *PS* isn't perfect, because it doesn't allow adversaries to issue delegation query oracles and proxy-sign query oracles for different proxy signer but same original signer. Second, we introduce an improved security model for *PS* which is stronger than the previous, and allows that an adversary queries the delegation query oracles and proxy-sign query oracles on different proxy signers possibly but the same original signer. The improved one overcomes the disadvantage of the previous. Third, we proposed a new paradigm *PS*-2-*CBS* which is a generic conversion from an existing secure *PS* to a secure *CBS*. That means we construct a new *CBS* scheme *PS*-2-*CBS* from an existing secure *PS* scheme. Table 1 lists the abbreviations and notations in the paper.

Table 1. Abbreviations and notations used in our work

Notation	Meaning
CBS	Certificate-based signature
PS	Proxy signature
PS-2-CBS	A conversion from a proxy signature to a certificate-based signature
C_{PS}	a challenger of the proxy signature
C_{CBS}	a challenger of the certificate-based signature
Π_{CB}	a certificate-based signature scheme
Π_{PS}	a proxy signature scheme

The rest of the paper is organized as follows. In Section 2, we give a brief summary about related syntax, adversarial types and security model of *PS*, and propose an improved security model for *PS*. We sketch the necessary definitions for *CBS* in Section 3. In Section 4, we introduce a generic conversion *PS*-2-*CBS* form the existing *PS*s to the *CBS*s, and prove that our *PS*-2-*CBS* is secure in the random oracle model. In Section 5, an example is given to illustrate the application of our new paradigm *PS*-2-*CBS*. Finally, we make a brief concluding remarks in Section 6.

2 The Proxy Signatures

We first review the syntax and security model for the *PS* [24], then analyse the drawback in existing security model of *PS*, and introduce an improved security model which is more perfect than the previous. For convenience, we use the prefix *PS*- to denote the *PS* system throughout the paper.

2.1 The Syntax of PS

A *PS* scheme delegated by warrant involves two entities, an original signer and a proxy signer, and is comprised of five algorithms including *PS-Setup*, *PS-KeyGen*, *PS-DeleGen*, *PS-PSign* and *PS-Verify*.

Definition 2.1 (*PS*). A proxy signature scheme delegated by warrant is defined as follows.

- *PS-Setup*(k): Takes input the system security parameter, and generates the system public parameters *PS-params*.
- *PS-KeyGen*(*PS-params*): Takes input the system public parameters, the algorithm generates the key pairs for signers. It includes two sub-algorithms as follows:
  - *PS-OKeyGen*(*PS-params*): Generates the original signer's private-public key (SK_O, PK_O).
  - *PS-PKeyGen*(*PS-params*): Generates the proxy signer's private-public key (SK_P, PK_P).
- *PS-DeleGen*(*PS-params*, w, SK_O): Takes input the system public parameters, the original signer's private key and a warrant, the algorithm generates a delegation D_w on the warrant w.
- *PS-PSign*(m, *PS-params*, w, D_w, SK_P): Takes input a message, the system public parameters, a warrant and its delegation, the proxy signer's private key, generates a proxy signature σ which corresponds to the message m.
- *PS-Verify*(m, σ, *PS-params*, w, D_w, PK_O, PK_P): Takes input the message/signature pair, the system public parameters, a warrant and its delegation, the original signer's and the proxy signer's public key. The algorithm returns "*accept*" if signature σ is a valid signature on the message m, otherwise returns "*reject*".

2.2 Security Model for PS

2.2.1 Adversarial Model

There are three types of adversaries with different capabilities in a *PS* scheme. A *PS* scheme is secure if it can resist each type of adversary.

- Adversary A_1: Type 1 Adversary A_1 has the private key of the proxy signer, and the public keys of the original signer and proxy signer, which simulates a malicious proxy signer.
- Adversary A_2: Type 2 adversary A_2 has the private key of the original signer, and the public keys of the original signer and proxy signer, which simulates a malicious original signer.
- Adversary A_3: Type 3 Adversary A_3 only has the public keys of original signer and proxy signer,

which simulates an outside adversary.

2.2.2 The Existing Attack Model

A *PS* scheme must be existential unforgeable against adversaries A_1, A_2 and A_3, respectively. It is obvious if a *PS* scheme is existential unforgeable against adversaries A_1 *and* A_2, then it must be existential unforgeable against adversary A_3. Therefore, we can only consider the existential unforgeable against type 1-2 adversaries for a *PS* scheme. The existential unforgeability of the *PS* is defined by the game1 and game2, in which the adversaries A_1 and A_2 will interact with their challenger, respectively.

Game 1. The existential unforgeability against a type 1 adversary A_1 is defined by the following game, in which the adversaries A_1 will interact with its challenger *C*.

- *PS-Setup*: The challenger *C* runs the algorithm *PS-Setup* to get the system public parameters *PS-params*, and runs the algorithm *PS-KeyGen* to get key pair of the original signer and the proxy signer: (SK_O, PK_O) and (SK_P, PK_P), returns *PS-params* and (SK_P, PK_P, PK_O) to the adversary A_1.
- *PS-Query* Oracles: In polynomial time, the adversary A_1 can request *DeleQuery* and *PSignQuery* oracles adaptively.
  (1) *DeleQuery*: On a new *DeleQuery*(w_i), the challenger *C* runs the algorithm *PS-DeleGen* to get delegation D_{wi}, and returns D_{wi} to the adversary A_1.
  (2) *PSignQuery*: On a new *PSignQuery*(m_j, w_i), the challenger *C* first issues the *PS-DeleQuery*(w_i) to obtain the delegation D_{wi} corresponding the warrant w_i, then runs the algorithm *PS-PSign*, and returns a signature σ_j on the message m_j to A_1.
- *PS-Output*: Adversary A_1 outputs a signature forgery σ^* finally, such that:
  - σ^* is a valid proxy signature on the message m^* under the warrant w^*;
  - w^* has never been submitted to *DeleQuery*;
  - (m^*, w^*) has never been submitted to *PSignQuery*.

Game 2. The existential unforgeability against a type 2 adversary A_2 is defined by the following game, in which the adversaries A_2 will interact with its challenger *C*.

- *PS-Setup*: The challenger *C* runs the algorithm *PS-Setup* to get the system public parameters *PS-params*, and runs the algorithm *PS-KeyGen* to get key pair of the original signer and the proxy signer: (SK_O, PK_O) and (SK_P, PK_P), returns *PS-params* and (SK_O, PK_O, PK_P) to the adversary A_2.
- *PS-Query* Oracles: In polynomial time, type 2 adversary A_2 can request *PSignQuery* oracles adaptively.
  - *PSignQuery*: On a new *PSignQuery*(m_j, w_i), the challenger *C* returns a signature σ_j on m_j to A_2.
- *PS-Output*: Adversary A_2 outputs a forged signature σ^* finally, such that:
  - σ^* is a valid proxy signature on the message m^* under the warrant w^*;
  - (m^*, w^*) has never been submitted to *PSignQuery*.

2.2.3 The Improved Attack Model

As mentioned, the existing security model of *PS* is only used in the case of fixed proxy signer and single one. That means, in the existing security model, adversary is neither allowed to query other proxy signers' delegation except the specified proxy signer, nor to query other proxy signers' signature except the specified proxy signer, and the forged proxy signature must also be the specified proxy signer's signature. Thus, the adversary attack model defined as above is not strong enough. We will introduce an improved adversary attack model which allows both queries as mentioned above. The improved attack model will overcome the drawback mentioned above and allows that an adversary queries the delegation query oracles and the proxy-sign query oracles for different proxy signers but always the same original signer.

Definition 2.2 (*PS-Game*1). The game is defined between a type 1 adversary A_1 and a challenger *C*.

- *PS-Setup*: For a given security parameter *k*, the challenger *C* runs the algorithm *PS-Setup* to obtain the system public parameters *PS-params*, and runs the algorithm *PS-KeyGen* to obtain the original signer *O*'s key pair (SK_O, PK_O), returns *PS-params* and PK_O to the adversary A_1.
- P*S-Query* Oracles: In polynomial time, the adversary A_1 can request *PKeyQuery*, *DeleQuery* and *PSignQuery* oracles adaptively.
  (1) *PKeyQuery*: Let P_i denote the identity of a proxy signer. On a new *PKeyQuery*(P_i), the challenger *C* returns the proxy signer P_i's key pair (SK_{Pi}, PK_{Pi}) to A_1.
  (2) *DeleQuery*: On a new *DeleQuery*(w_j, P_i), the challenger *C* returns a delegation D_{wj} to A_1.
  (3) *PSignQuery*: On a new *PSignQuery*(m, w_j, P_i), the challenger *C* returns a signature σ to A_1.
- *PS-Output*: Adversary A_1 outputs a forged proxy signature σ^* on the message m^* under warrant w^* for the proxy signer P^* finally, such that:
  - σ^* is a valid proxy signature on the message m^* under the warrant w^* and the proxy signer P^*;
  - (w^*, P^*) has never been submitted to *DeleQuery*;
  - (m^*, w^*, P^*) has never been submitted to *PSignQuery*.

Definition 2.3 (*PS-Game* 2). The game is defined between a type 2 adversary A_2 and a challenger *C*.

- *PS-Setup*: For a given security parameter k, the challenger C runs the algorithm *PS-Setup* to obtain the system public parameters *PS-params*, and runs the algorithm *PS-OKeyGen* to obtain the original signer O's key pair (SK_O, PK_O), gives *PS-params* and (SK_O, PK_O) to the adversary A_2.
- *PS-Query* Oracles: In polynomial time, the adversary A_2 can request *PKeyQuery*, *ReleaseQuery* and *PSignQuery* oracles adaptively.
  (1) *KeyQuery*: Let P_i denote the identity of a proxy signer. On a new *PKeyQuery*(P_i), the challenger C returns the proxy signer P_i's public key PK_{Pi} to A_2.
  (2) *ReleaseQuery*: On a new *ReleaseQuery*(P_i), the challenger C returns P_i's private key SK_{Pi} to A_2.
  (3) *PSignQuery*: On a new *PSignQuery*(m, w_j, P_i), the challenger C returns a signature σ to A_2.
- *PS-Output*: Adversary A_2 outputs a forged signature σ^* finally, such that:
  - σ^* is a valid proxy signature on the message m^* under the warrant w^* and the proxy signer P^*;
  - P^* has never been submitted to *ReleaseQuery*;
  - (m^*, w^*, P^*) has never been submitted to *PSignQuery*.

Definition 2.4 (Unforgeability of *PS*). A proxy signature scheme is existential unforgeable under adaptively chosen message attacks iff the probability of success that any polynomial bounded adversary A_1 and A_2 win the *PS-Game* 1 and *PS-Game* 2 respectively is negligible.

3 The Certificate-based Signatures

We review the definitions of *CBS* [17], and use the prefix *CB-* to denote a *CBS* system in the paper.

3.1 The Syntax of CBS

Definition 3.1 (*CBS*). A certificate-based signature scheme involves two entities, a *CA* and a signer, and is comprised of five algorithms.

- *CB-Setup*(k): Takes input a security parameter, and generates the *CA*'s master key pair (*mpk*, *msk*) and the system public parameters *CB-params*.
- *CB-UKeyGen*(*CB-params*, *ID*): Takes input the system public parameters and the signer's identity, generates (PK_{ID}, SK_{ID}) as the signer's public/private key.
- *CB-CertGen*(*CB-params*, *msk*, *ID*, PK_{ID}): Takes input the system public parameters, the *CA*'s master secret key, the signer's identity and his public key, generates a signer's certificate $Cert_{ID}$.
- *CB-Sign*(m, *CB-params*, *ID*, SK_{ID}, $Cert_{ID}$): Takes input a message, the system public parameters, the signer's identity and his private key, certificate, generates a signature σ which corresponds to the message m.
- *CB-Verify*(m, σ, *CB-params*, *mpk*, *ID*, PK_{ID}): Takes input a message/*CBS* pair, the system public parameters, the *CA*'s master public key, the signer's identity and his public key, outputs "accept" if σ is valid signature, otherwise, outputs "reject".

3.2 Security Model of CBS

There are two types of adversaries with different capabilities, A_I and A_{II}. A *CBS* scheme must be secure against each type of adversaries. The type I adversary A_I simulates the scenario where the adversary is allowed to replace public keys of any entities except the certifier, and A_I is in possession of the private key of the signer, but doesn't know anything about the *CA*'s master secret key. The type *II* adversary A_{II} simulates a malicious *CA* which is able to produce certificate but is not allowed to replace the target signer's public key, and doesn't know anything about the signer's private key. The unforgeability of the *CBS* is defined by two games *CB-Game* 1 and *CB-Game* 2, in which A_I and A_{II} will interact with their challenger C, respectively.

Definition 3.2 (*CB-Game* 1). The *CB-Game* 1 is defined by the following game.

- *CB-Setup*: The challenger C runs *CB-Setup*(k), returns the system public parameters *CB-params* and the system master public key *mpk* to the adversary A_I, and keeps the system master secret key *msk* by himself.
- *CB-Query* Oracls: In polynomial time t, the adversary A_I issues query oracles as follows:
  (1) *UKeyQuery*. On a new *UKeyQuery*(ID_i), if ID_i has already been created, nothing is to be performed by the challenger C, otherwise, the C runs *CB-UKeyGen* and returns ID_i's key pair (SK_{IDi}, PK_{IDi}) to A_I .
  (2) *CertQuery*. On a new *CertQuery*(ID_i, PK_{IDi}), the challenger C returns a certificate $Cert_{IDi}$ to A_I.
  (3) *ReplPKQuery*. On a new *ReplPKQuery*(ID_i), the adversary A_I replaces ID_i 's public key with a new value PK'_{IDi} which is chose by himself.
  (4) *SignQuery*. On a new *SignQuery*(m, ID_i, PK_{IDi}), C runs *CB-Sign* and returns a signature σ to A_I.
- *CB-Output*: Adversary A_I outputs a signature forgery σ^* finally such that:
  - σ^* is a valid signature on the message m^* under the public key $PK_{ID}{}^*$ with the identity ID^*;
  - (ID^*, $PK_{ID}{}^*$) has never been submitted to *CertQuery* oracle;
  - (m^*, ID^*, $PK_{ID}{}^*$) has never been submitted to *SignQuery* oracle.

Definition 3.3 (*CB-Game* 2). The *CB-game* 2 is defined by the following game.

- *CB-Setup*: The challenger C runs the algorithm *CB-Setup*(k), returns the system public parameters *CB-*

params and the system master key pair (*mpk*, *msk*) to the adversary A_{II}.

- *CB-Query* Oracles: In polynomial time *t*, the adversary A_{II} can adaptively issue the *UKeyQuery*, *CorruptionQuery* and *SignQuery* oracles, but doesn't issue *CertQuery* oracles, because A_{II} has the knowledge of the *CA*'s master secret key *msk* and he can generate the signer's certificate.
  (1) *UKeyQuery*. On a new *UKeyQuery*(ID_i), the challenger *C* runs the algorithm *CB-UKeyGen* and returns ID_i's public key PK_{IDi} to A_{II}.
  (2) *CorruptionQuery*. On a new *CorruptionQuery*(ID_i), the challenger *C* returns ID_i's private key SK_{IDi} to A_{II} if ID_i has been created.
  (3) *SignQuery*. The *SignQuery* is similar to *CB-Game* 1.
- *CB-Output*: Adversary A_{II} outputs a signature forgery σ^* finally such that:
  - σ^* is a valid signature on the message m^* under the public key $PK_{ID}{}^*$ with the identity ID^*;
  - ID^* has never been submitted to *CorruptionQuery* oracle.
  - (m^*, ID^*) has never been submitted to *SignQuery* oracle.

Definition 3.4 (Unforgability of *CBS*). If and only if the probability is negligible that any polynomial bounded adversary A_I and A_{II} win the two games defined above, then a *CBS* scheme is existential unforgeable under adaptively chosen message attack.

4 The Generic Conversion from PS to CBS

We are aware of the common between *CBS* and *PS* through analyzed the similarities and differences between *CBS* and *PS*, and present a generic conversion *PS*-2-*CBS* from an existing *PS* to a *CBS*, and prove its security.

4.1 Comparisons

PS and *CBS* are completely different signature and are developed independently, but We find there are something in common between them.

First, there are two participants either in a *CBS* or a *PS* scheme. That is, there are a *CA* and a signer in a *CBS* scheme, and the *CA* generates an up-to-date certificate which corresponds a signer's identity and public key, while there are an original signer and a proxy signer in a *PS* scheme, and the original signer generates an authorization information which contains the signers' identity and scope of proxy signing and the valid period. Secondly, the action of two participants in a *CBS* is similar to that in a *PS*. More specifically, the *CA* in a *CBS* is similar to the original signer in a *PS*, they will both generate an authorization for another signer. That is, a delegation for the proxy signer in a *PS* or a certificate for the signer in a *CBS*, the signer in a *CBS* is similar to the proxy signer in a *PS*. They will both generate a valid signature by using authorization information and their own private key. In which, the authorization information is a certificate in CBS and a delegation in PS. Thirdly, either the *CBS* or the *PS*, two pieces of secret information are required when generating a signature. That is, it will require both a proxy signer's private key and a delegation when generating the *PS* on a message, while it will require both a signer's private key and a certificate when generating the *CBS* on a message.

4.2 The Conversion from PS to CBS

We introduce a generic conversion from a secure *PS* to a secure *CBS* to construct a *PS*-2-*CB*S below. We will use Π_{PS} to denote a *PS* scheme, and Π_{CB} to denote a *CBS* scheme below.

- *CB-Setup*: Takes inputting a security parameter *k*, runs *PS-Setup*(*k*) of Π_{PS} to get *PS-params*, then runs *OKeyGen*(*PS-params*) of Π_{PS} to get (SK_O, PK_O). Sets *CB-params*=*PS-params*, $mpk=PK_O$, $msk=SK_O$. Returns *CB-params* as the system public parameters and (*mpk*, *msk*) as the system master key pair of Π_{CB}.
- *CB-UKeyGen*: Takes inputting the system public parameters *CB-params* and the signer's identity *ID*, sets *PS-params*=*CB-params*, runs *PKeyGen*(*PS-params*) of Π_{PS} to get (SK_P, PK_P), and sets (SK_{ID}, PK_{ID})=(SK_P, PK_P). Returns (SK_{ID}, PK_{ID}) as the signer *ID*'s key pair of Π_{CB}.
- *CB-CertGen*: Takes inputting the system public parameters *CB-params* and the system maser secret key *msk*, a signer's identity *ID* and his public key PK_{ID}, sets *PS-params*=*CB-params*, $w=ID||PK_{ID}$, $SK_O=msk$, runs *PS-DeleGen*(*PS-params*, *w*, SK_O) of Π_{PS} to get D_w, then sets $Cert_{ID}=D_w$. Returns $Cert_{ID}$ as the signer *ID*'s certificate.
- *CB-PSign*: Takes inputting a message *m* to be signed, the system public parameters *CB-params*, a signer's identity *ID* and his private key SK_{ID}, certificate $Cert_{ID}$, sets *PS-params*=*CB-params*, $w=ID||PK_{ID}$, $D_w=Cert_{ID}$, $SK_P=SK_{ID}$, runs *PS-PSign*(*m*, *PS-params*, *w*, D_w, SK_P) of Π_{PS} to gets a signature σ. Returns σ as a *CBS* on *m*.
- *CB-Verify*: Takes inputting a message *m* and the corresponding signature σ, public parameters *CB-params*, the master public key *mpk*, a signer's identity and public key pair (*ID*, PK_{ID}), sets *PS-params*=*CB-params*, $w=ID||PK_{ID}$, $D_w=Cert_{ID}$, $PK_O=mpk$, $PK_P=PK_{ID}$. Returns *PS-Verify*(*m*, σ, *PS-params*, *w*, D_w, PK_O, PK_P).

4.3 Security Proof

Theorem 1 (Unforgeability). The constructed *PS*-2-*CBS* scheme is existential unforgeable against adaptively chosen-message attack if the underlying *PS* scheme is secure in improved security model of *PS*.

Lemma 1. The proposed *PS*-2-*CBS* scheme is existential unforgeable against type *I* adversary *CB-A*$_I$ if the underlying *PS* scheme is existentially unforgeable against type 1 adversary *PS-A*$_1$ under adaptively chosen-message attack in improved security model of *PS*.

Proof: We denote a type *I* adversary of *CBS by CB-A*$_I$. Assume that *CB-A*$_I$ can win *CB-Game* 1 of *PS*-2-*CBS*, then we can construct a type 1 adversary *PS-A*$_1$ to win the *PS-Game* 1 for underlying *PS* scheme, in which, *PS-A*$_1$ is the challenger C_{CB} simultaneously. We denote a challenger of the *PS* by C_{PS}.

- *CB-Setup*: The challenger C_{PS} first runs *PS-Setup*(*k*) of Π_{PS} to obtain *PS-params*, then runs *PS-OKeyGen*(*PS-params*) of Π_{PS} to get (SK_O, PK_O), returns {*PS-params*, PK_O} to *PS-A*$_1$. *PS-A*$_1$ sets *CB-params=PS-params*, $mpk=PK_O$, returns {*CB*−*params*, *mpk*} to *CB-A*$_I$.
- *CB-Query* Oracles: Type *I* adversary *CB-A*$_I$ issue the following query oracles adaptively:
  - *UKeyQuery*: For a new query ID_i, type I adversary *CB-A*$_I$ gives ID_i to *PS-A*$_1$, *PS-A*$_1$ sets $P_i=ID_i$, and sends to the challenger C_{PS}. The challenger C_{PS} issues the *PKeyQuery*(P_i), and returns P_i's key pair (SK_{Pi}, PK_{Pi}) to *PS-A*$_1$; *PS-A*$_1$ sets (SK_{IDi}, PK_{IDi}) = (SK_{Pi}, PK_{Pi}), returns (SK_{IDi}, PK_{IDi}) to *CB-A*$_I$.
  - *CertQuery*: For a new query (ID_i, PK_{IDi}), type I adversary *CB-A*$_I$ gives (ID_i, PK_{IDi}) to *PS-A*$_1$, *PS-A*$_1$ sets $w_i=ID_i||PK_{IDi}$, $P_i=ID_i$ and sends to the challenger C_{PS}. The challenger C_{PS} issues *DeleQuery*(w_i, P_i), and returns D_{wi} to *PS-A*$_1$; *PS-A*$_1$ sets $Cert_{IDi}=D_{wi}$, and returns $Cert_{IDi}$ to *CB-A*$_I$.
  - *ReplPKQuery*: When *CB-A*$_I$ makes the query on (ID_i, PK'_{IDi}), C_{CB} sets PK'_{IDi} as the current public key.
  - *SignQuery*: For a new query (*mj*, ID_i, PK_{IDi}), type I adversary *CB-A*$_I$ sends (m_j, ID_i, PK_{IDi}) to *PS-A*$_1$, *PS-A*$_1$ sets $w_i=ID_i||PK_{IDi}$, $D_{wi}=Cert_{IDi}$, $P_i=ID_i$ and sends to the challenger C_{PS}. The challenger C_{PS} issues *PSignQuery*(m_j, w_i, P_i) to obtain a signature σ_j, and returns σ_j to *PS-A*$_1$; *PS-A*$_1$ returns σ_j to *CB-A*$_I$.
- *CB-Output*: Finally, *CB-A*$_I$ outputs a forged *CBS* σ^* on m^* for a target ID^* and $PK_{ID}{}^*$. *CB-A*$_I$ sets $w^*=ID^*||PK_{ID}{}^*$, $P^*=ID^*$, outputs (m^*, σ^*, w^*) as a *PS* forgery. If σ^* is a valid *CBS* forgery for a target ID^* and $PK_{ID}{}^*$, then σ^* must be a valid *PS* under the warrant w^* and the proxy signer P^*. This means that if we forge a *CBS* signature σ^*, then σ^* must be a forgery of *PS*, and our *PS*-2-*CBS* scheme is existentially unforgeable against type *I* adversary *CB-A*$_I$ if underlying *PS* scheme is existentially unforgeable against type 1 adversary *PS-A*$_1$ in improved security model of *PS*. The proof process is illustrated in Figure 1.

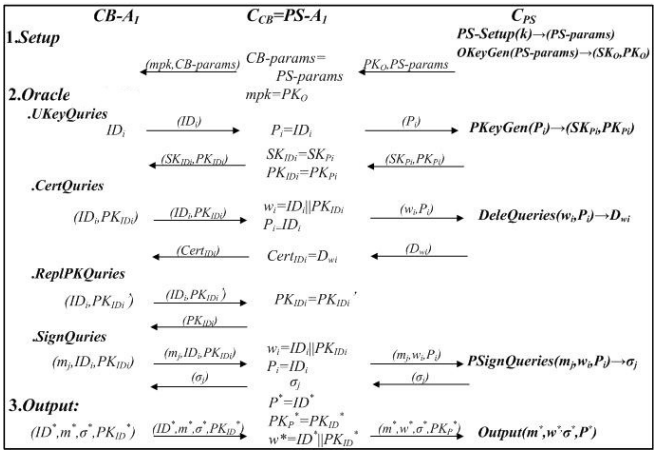

Figure 1. Proof diagram of *PS*-2-*CBS* Game 1

Lemma 2. The proposed *PS*-2-*CBS* scheme is existential unforgeable against type *II* adversary *CB-A*$_{II}$ if the underlying *PS* scheme is existentially unforgeable against type 2 adversary *PS-A*$_2$ under adaptively chosen-message attack in improved security model of *PS*.

Proof: We denote a type *II* adversary of *CBS by CB-A*$_{II}$. Assume that *CB-A*$_{II}$ can win *CB-Game* 2 of *PS*-2-*CBS* above, then we can construct a type 2 adversary *PS-A*$_2$ to win the *PS-Game* 2 for underlying *PS* scheme, in which, *PS-A*$_2$ is the challenger C_{CB} simultaneously. We denote a challenger of the *PS* by C_{PS}.

- CB-Setup: The challenger C_{PS} first runs PS-Setup(k) of Π_{PS} to obtain PS-params, then runs PS-OKeyGen (PS-params) of Π_{PS} to get (SK_O, PK_O), sends PS-params, SK_O, PK_O to PS-A$_2$. PS-A$_2$ sets CB-params=PS-params, $mpk=PK_O$, $msk=SK_O$, and returns CB-params, mpk, msk to CB-A$_{II}$.
- *CB-Query* Oracles: Type *II* adversary *CB-A*$_{II}$ issues adaptively query racles as follows.
  - *UKeyQuery*: For a new query ID_i, the adversary *CB-A*$_{II}$ gives ID_i to *PS-A*$_2$, *PS-A*$_2$ sets $P_i=ID_i$ and sends to the challenger C_{PS}. The challenger C_{PS} issues *PKeyQuery*(P_i) and returns P_i's public key PK_{Pi} to *PS-A*$_2$; *PS-A*$_2$ sets $PK_{IDi}=PK_{Pi}$, returns PK_{IDi} to *CB-A*$_{II}$.
  - CorruptionQuery: For a new query ID_i, the adversary *CB-A*$_{II}$ gives ID_i to *PS-A*$_2$, *PS-A*$_2$ sets $P_i=ID_i$ and sends to the challenger C_{PS}. The challenger C_{PS} issues *ReleaseQuery*(P_i) and returns the proxy signer P_i's private key SK_{Pi} to *PS-A*$_2$; *PS-A*$_2$ sets $SK_{IDi}=SK_{Pi}$, and returns SK_{IDi} to *CB-A*$_{II}$.
  - *SignQuery*: For a new query (m_j, ID_i, PK_{IDi}), the adversary *CB*−*A*$_{II}$ sends (m_j, ID_i, PK_{IDi}) to *PS-A*$_2$, *PS-A*$_2$ sets $w_i=ID_i||PK_{IDi}$, $D_{wi}=Cert_{IDi}$, $P_i=ID_i$ and sends to the challenger C_{PS}. The challenger C_{PS} issues *PSignQuery* (m_j, w_i, P_i) to obtain a signature σ_j, and returns σ_j to *PS-A*$_2$; *PS-A*$_2$

returns σ_j to $CB\text{-}A_{II}$.

- *CB-Output*: $CB\text{-}A_{II}$ outputs a forged *CBS* σ^* on the m^* for a target ID^* and the public key $PK_{ID}{}^*$ finally. $CBS\text{-}A_{II}$ sets $w^*=ID^*||PK_{ID}{}^*$, $P^*=ID^*$, outputs (m^*, σ^*, w^*) as a *PS* forgery. If σ^* is a valid *CBS* forgery for a target ID^* and $PK_{ID}{}^*$, then σ^* must be a valid *PS* under the warrant w^* and the proxy signer P^*. This means that if we forge a *CBS* signature σ^* successfully, then the signature σ^* must be a forgery for *PS*. The proposed *PS*-2-*CBS* scheme is existentially unforgeable against type *II* adversary $CB\text{-}A_{II}$ if underlying *PS* scheme is existentially unforgeable against type 2 adversary $PS\text{-}A_2$ in improved security model of *PS*. The proof process is illustrated in Figure 2.

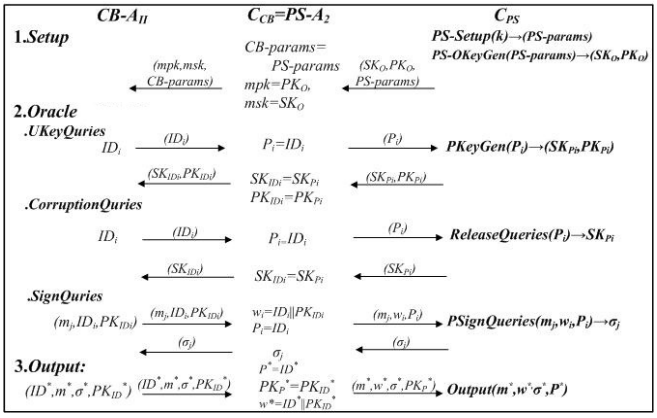

Figure 2. Proof diagram of *PS*-2-*CBS* Game 2

5 An Example of PS-2-CBS

We give a concrete example of the *PS*-2-*CBS*. We first sketch out an existing *PS* scheme [25], and construct a concrete *CBS* scheme by using our generic construction *PS*-2-*CBS* based on the scheme [25].

5.1 Underlying PS Scheme

The [25]'s proxy signature scheme consists of the following algorithms.

- *Setup*: Let k be the system security parameter, G_1 be an additive group with prime order q, $P \in G_1$ is a generator, and G_2 be a multiplicative group of the same order q. A bilinear pairing is a map $e: G_1 \times G_1 \rightarrow G_2$. H_0: $\{0, 1\}^* \rightarrow G_1$ and H_1: $\{0, 1\}^* \rightarrow Z_q^*$ are two distinct cryptographic hash functions. The system public parameters are $params=\{k, G_1, G_2, e, q, P, H_0, H_1\}$.
- *KeyGen*: The original signer A picks $x_A \in Z_q^*$ at random, outputs the original signer A's key pair $(x_A, P_A=x_AP)$. The proxy signer B picks $x_B \in Z_q^*$ at random, outputs the proxy signer B's key pair $(x_B, P_B=x_BP)$.
- *DeleGen*:
  - Given a warrant w, the original signer A computes $Q_B=H_0(ID_B, P_B, w)$, $D_{AB}=x_AQ_B$, where ID_B is the identity of proxy signer B. Output D_{AB} as a delegation under the warrant w;
  - The original signer A sends D_{AB} to the proxy signer B;
  - The proxy signer B verifies whether the equation holds: $e(D_{AB}, P)=e(Q_B, P_A)$;
  - The proxy signer B sets (x_B, D_{AB}) as his proxy key.
- *Sign*: Given a message m to be signed, the proxy signer B compute $\sigma=(1\div(H_1(m)+x_B))D_{AB}$. Outputs σ as a proxy signature on the message m.
- *Verification*: Given a message/signature pair (m, σ), the system public parameters *params*, the original signer A's public key P_A and the proxy signer B's public key P_B, the algorithm checks the equation $e(\sigma, H_1(m)P + P_B)=e(Q_B, P_A)$. If the equality holds, outputs "*accept*", otherwise, outputs "*reject*".

5.2 The Converted PS-2-CBS Scheme

We produce a *CBS* scheme from the *PS* which is illustrated in section 5.1 by using the *PS*-2-*CBS*. The produced certificated-based signature *PS*-2-*CBS* scheme is as follows.

- *Setup*: It is the same as in the Section 5.1 for generating the system parameters. The algorithm picks $s_C \in Z_q^*$ at random, and sets $msk=s_C$ as the system master secret key, computes $mpk=s_CP$ as the system master public key. The system public parameters are $params=\{k, G_1, G_2, e, q, P, H_0, H_1\}$.
- *UKeyGen*: Given the system public parameters *params*, the system master public key *mpk* and a signer's identity ID_A, the algorithm picks $s_A \in Z_q^*$ at random, sets $SK_A=s_A$ and computes $PK_A=s_AP$, then the signer ID_A's key pair is (SK_A, PK_A).
- *CertGen*: Given the system public parameters *params*, the system master secret key *msk*, a signer's identity ID_A and his public key PK_A, the algorithm computes $Q_A=H_0(ID_A, PK_A, ID_A||PK_A)$ and $Cert_A=s_CQ_A$, which can be verified by checking the equation: $e(Cert_A, P)=e(Q_A, mpk)$.
- *Sign*: Given a message m to be signed, the system public parameters *params*, the system master public key *mpk*, a signer's identity ID_A and his public key PK_A. The signer works as follows:
  - The temporary signing key is $S_A=(s_A, Cert_A)$;
  - Computes $h=H_1(m)$, $\sigma=(1\div(h+s_A))Cert_A$.

  Outputs σ as a certificated-based signature on the message m.
- Verifiy: Given a message/signature pair (m, σ), the system public parameters *params*, the system master public key *mpk*, and a signer's identity ID_A and his public key PK_A, the algorithm works as follows:
  - Computes $Q_A=H_0(ID_A, PK_A, ID_A||PK_A)$, $h=H_1(m)$;

- Checks whether the equation $e(\sigma, hP+PK_A)=e(Q_A, mpk)$ holds. If it holds, outputs "*accept"*, otherwise, outputs "*reject"*.

6 Conclusion

In this paper, aiming at constructing a generic conversion *PS*-2-*CBS* from *PS* to *CBS*, we introduced an improved security model of *PS* after analyzed the drawback of the existing one. In contrast to existing security model, improved one is stronger and allows an adversary of *PS* access to delegation queries and proxy-sign queries for different proxy signers but the same original signer. With the help of the improved security model, we proposed a new paradigm *PS*-2-*CBS* which is a generic conversion from an existing secure *PS* to a secure *CBS*. With the aid of the *PS*-2-*CBS*, we can construct a *CBS* conveniently by using an existing *PS*. Comparing with traditional *PKI*-based system and the identity-based system, the certificate-based signature simplifies use and management of certificates, and overcomes key escrow problem well. In the future, we will try to research the relationship between the special *PS* and special *CBS*, and the convertion of them.

Acknowledgements

The authors acknowledge the Natural Science Foundation of Fujian Province of China (Grant: 2019J01750), the National Science Foundation of China (Grant: 61170246), the Education and Scientific Research Fund for Young and Middle-aged Teachers of Fujian Province of China (Grant: JA170345). Besides, the authors would like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[1] M. Mambo, K. Usuda, E. Okamoto, Proxy signatures: delegation of the power to sign messages, *Proceedings of IEICE transactions on fundamentals of electronics, communications and computer sciences*, Vol. E79-A, No 9, pp. 1338-1354, September, 1996.

[2] J. Dai, X. Yang, J. Dong, Designated-receiver proxy signature scheme for electronic commerce, *International Conference on Systems, Man and Cybernetics*, Washington, DC, 2003, pp. 384-389.

[3] B. C. Neuman, Proxy-based authorization and accounting for distributed systems, *13th International Conference on Distributed Computing Systems*, Pittsburgh, Pennsylvania, 1993, pp. 283-291.

[4] S. Kim, S. Park, D. Won, Proxy Signatures, Revisited, *Information and Communications Security*, Beijing, China, 1997, pp. 223-232.

[5] V. Varadharajan, P. Allen, S. Black, An analysis of the proxy problem in distributed systems, 1991 *IEEE computer society symposium on research in security and privacy*, Oakland, CA, 1991, pp. 255-275.

[6] H. Q. Wang, A Proxy Ring Signature Scheme with Revocable Anonymity Based on Bilinear Pairings, *Journal of Internet Technology*, Vol. 11, No. 5, pp. 627-632, September, 2010.

[7] J. H. Zhang, P. Li, On the Security of a Proxy Ring Signature with Revocable Anonymity, *Journal of Internet Technology*, Vol. 16, No. 7, pp. 1169-1175, December, 2015.

[8] G. B. Xu, Novel Quantum Proxy Signature without Entanglement, *International Journal of Theoretical Physics*, Vol. 54, No. 8, pp. 2605-2612, August, 2015.

[9] H. W. Qin, K. S. Wallace, Tang, T. Raylin, Batch quantum multi-proxy signature, *Optical and quantum electronics*, Vol. 50, No. 12, pp. 450.1-450.8, November, 2018.

[10] F. Zhu, X. Tao, C. Lin, W. Wu, A Proxy Transitive Signature Scheme, *Journal of Internet Technology*, Vol. 19, No. 4, pp. 1273-1284, July, 2018.

[11] C. Zhou, Z. Cui, G. Gao, On the Security of an Improved Identity-based Proxy Signature Scheme without Random Oracles, *Journal of Internet Technology*, Vol. 19, No. 7, pp. 2057-2068, December, 2018.

[12] F. G. Wu, Y. Wang, Z. Xiao, Z. M. Zheng, An Efficient Lattice-Based Proxy Signature with Message Recovery, 2017 *Security, Privacy, and Anonymity in Computation*, *Communication*, and Storage: 10th International Conference (SpaCCS 2017), Guangzhou, China, 2017, pp. 321-331.

[13] C. X. Sun, Y. F. Guo, Y. L. Li, One Secure Attribute-Based Proxy Signature, *Wireless Personal Communications*, Vol. 103, No. 2, pp. 1273-1283, November, 2018.

[14] C. Gentry, Certificate-based Encryption and the Certificate Revocation Problem, *International Conference on the Theory and Applications of Cryptographic Techniques*, Warsaw, Poland, 2003, pp. 272-293.

[15] B. G. Kang, J. H. Park, S. G. Hahn, A Certificate-based Signature Scheme, *Cryptographers' Track at the Rsa Conference*, San Francisco, CA, 2004, 99-111.

[16] A. Shamir, Identity-Based Cryptosystems and Signature Schemes, Workshop on the Theory and Application of Cryptographic Techniques, in: G. R. Blakley, D. Chaum (Eds), Advances in Cryptology. CRYPTO 1984. *Lecture Notes in Computer Science*, vol 196, Springer, Berlin, Heidelberg, 1985, pp. 47-53.

[17] J. G. Li, X. Y. Huang, Y. Mu, S. Willy, Q. H. Wu, Certificate-based Signature: Security Model and Efficient Construction, *Public Key Infrastructure: 4th European PKI Workshop*, Palma de Mallorca, Spain, 2007, pp. 110-125.

[18] M. H. Au, J. K. Liu, W. Susilo, T. H. Yuen, Certificate Based (Linkable) Ring Signature, in: E. Dawson, D. S. Wong (Eds.), *Information Security Practice and Experience*. ISPEC 2007. *Lecture Notes in Computer Science*, vol. 4464, Springer, Berlin, Heidelberg, 2007, pp. 79-92.

[19] G. K. Verma, B. B. Singh, H. Singh, Provably Secure Certificate-Based Proxy Blind Signature Scheme from Pairings, *Information Sciences*, Vol. 468, pp. 1-13, November,

2018.

[20] J. Li, Z. Wang, Y. Zhang, Provably secure certificate-based signature scheme without pairings, *Information Sciences*, Vol. 233, pp. 313-320, June, 2013.

[21] R. F. Huang, Q. Nong, Efficient Certificate-Based Blind Signature Scheme without Bilinear Pairings, *Applied Mechanics and Materials*, Vol. 220-223, pp. 2735–2739, November, 2012.

[22] X. X. Ma, J. Shao, C. Zuo, R. Meng, Efficient Certificate-Based Signature and Its Aggregation, in: J. Liu, P. Samarati (Eds.), Information Security Practice and Experience. ISPEC 2017. *Lecture Notes in Computer Science*, vol. 10701, Springer, Cham, 2017, pp. 391-408.

[23] R. F. Huang, Z. J. Huang, Q. S. Chen, Provable Secure Generic Construction of Proxy Signature from Certificate-based Signature, *The Open Automation and Control Systems Journal*, Vol. 6, No. 1, pp. 566-574, December, 2014.

[24] Y. Yong, Y. Mu, S. Willy, Y. Sun, Y. F. Ji, Provably secure proxy signature scheme from factorization, *Journal of Mathematical and Computer Modelling*, Vol. 55, No. 3-4, pp. 1160-1168, February, 2012.

[25] X. Y. Huang, Y. Mu, S. Willy, F. G. Zhang, X. F. Chen, A Short Proxy Signature Scheme: Efficient Authentication in the Ubiquitous World, *International Conference on Embedded and Ubiquitous Computing (EUC 2005)*, Nagasaki, Japan, 2005, pp. 480-489.

Biographies

Rufen Huang received his M.S. from Department of Automation at Xiamen University. She is an professor in the School of Computer Science at Minnan Normal University, China. Her research interests include Network, cryptography and information security. She has published research papers in the area of cryptography and information security.

Zhenjie Huang received the Ph.D. degree in cryptography from Xidian University, China in 2005. He is currently a full professor in the Fujian Key Laboratory of Granular Computing and Application, Minnan Normal University. His current research interests include cryptography and information security.

Qunshan Chen received his M.S. in Mathematics from Xiamen University. He is a lecturer in the School of Computer Science at Minnan Normal University, China. His main research interests include cryptography and information security. His main contribution is in the area of digital signatures. He has published several research papers in the area of digital signatures.