Provably Secure Generic Construction of Certificate Based Signature from Certificateless Signature in Standard Model

Wei Gao[a,b], Guilin Wang[c], Kefei Chen[a], Xueli Wang[d]

[a] *Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China*
[b] *School of Mathematics and Information, Ludong University, Yantai 264025, China*
[c] *School of Computer Science & Software Engineering, University of Wollongong, NSW 2522, Australia*
[d] *School of Mathematics, South China Normal University, Guangzhou 510631, China*

Abstract

Similar to certificateless cryptography, certificate-based cryptography is another novel public key setting which combines the merits of traditional public key cryptography (PKC) and identity-based cryptography (IBC). Both certificateless cryptography and certificate based cryptography do not have the key escrow problem in identity-based cryptography, and greatly reduce the overhead of certificate management in traditional PKC. Public key encryption and digital signature are two main issues in public key cryptography. In this paper, we deal with the generic construction of certificate based signature from certificateless signature. In 2009, Wu et al. proposed the first generic conversion from certificateless signature (CLS) to certificate based signature (CBS). However, Wu et al.'s method has to involve a cryptographic hash function which is assumed to be a random oracle for security proof. This paper proposes another intuitive, simple and provably secure generic conversion from CLS to CBS. A new security model for CLS is formalized and then it is proved that this generic conversion is secure in the stand model, only if the underlying CLS scheme is secure in our new security model. As

Email addresses: `sdgaowei@gmail.com` (Wei Gao), `guilin@uow.edu.au` (Guilin Wang), `kfchen@sjtu.edu.cn` (Kefei Chen), `wangxuyuyan@gmail.com` (Xueli Wang)

[1]This work is completed when the first author visited University of Wollongong (2011-2012)

an example, based on this new generic method, we construct a new provably secure certificate based signature scheme.

Keywords: certificateless signature, certificate based signature, identity based signature, provable security.

---

1. Introduction

1.1. Background

Traditional VS Identity-based PKC (IBC). In traditional public key cryptography (PKC) the authenticity of the public keys must be certified by a trusted third party, which is called Certification Authority (CA). The public key infrastructure (PKI) required to support traditional PKC is the main difficulty in its deployment. Many of the problems of PKI-based cryptography arise from the management of certificates, which includes storage, revocation and distribution. In 1984, Shamir [7] proposed the concept of identity-based PKC (IB-PKC), which sought to reduce the requirements on the public key infrastructure by using the client's well-known identity as its public key. With this approach, the certification of public keys becomes implicit. However, because the private key of every user is generated and hence known by the authority called PKG, IB-PKC suffers from the main drawback of being inherently key escrowed, which limits the applicability of IB-PKC.

Certificate-based Cryptography (CBC). Motivated by the above problems of the traditional PKC and IB-PKC, the concepts of certificate based encryption (CBE) and certificate based signature (CBS) were introduced by Gentry [3] and Kang et al. [6] respectively. In PKC or IB-PKC, for decrypting or signing, the user only needs to know the private key. In certificate based cryptography (CBC), for decrypting/signing, the party Bob needs two indispensable data, namely his private key and the up-to-date certificate. In particular, this certificate acts as a partial decryption/signing key as well as a traditional public key certificate. Hence, before the user gets the up-date certificate, the user with only the private key remains unable to decrypt ciphertexts or generate signature. On one hand, CBC mitigates the problem of revocation of certificates, which is an especially challenging problem in certificate management in PKC. Now let the certificate have an expiry date, at the end of which a new certificate must be obtained from CA for decryption. In this way, revocation is achieved by stopping the issuance of certificates for the revoked public key. Hence this approach can greatly reduce the public key and certificate management cost. In other hand, there is no key-escrow problem, since the user's decryption/signing key is generated by joining both the certificate and a private information only known to the user. Additionally, there is no secret key distribution problem of IBE, since the CA's certificate need not be kept secret.

Certificateless cryptography (CLC). Similarly motivated by the disadvantages of the traditional PKC and IB-PKC, independently from the work of certificate based cryptography, the concept of certificateless encryption (CLE) and certificateless signature (CLS) were introduced by Al-Riyami and Paterson in [1]. In CLC, each entity has two secrets: a secret value SV chosen by the entity and a partial private key PPK generated by a third party called Private Key Generating(PKG). The full private key is the output of a function by taking SV and PPK as the input, and hence can be known by only the user. On one hand, unlike IBC, CLC does not suffer from key escrow, since PKG does not have access to the user's secret value SV. On the other hand, unlike traditional PKC, CLC does not require the use of certificates to guarantee the authenticity of public keys, because any outside attacker not being PKG is not able to figure out the partial private key PPK for the authentic or fake public key.

1.2. Related Works.

Although CBC and CLC were developed independently, both of them can be conceptually seen as intermediates between traditional PKC and IBC, seeking to simplify the management of certificates while avoiding the key escrow property of identity-based cryptography. Of course, the main difference between them is whether there are certificates required. So a natural question to establish the connection of these two concepts arose. In 2005, Al-Riyami and Paterson in [2] presented a generic construction of a secure CBE scheme from a secure CLE scheme and claimed the provable security of this generic construction of CBE. Shortly later, Kang and Park [5] pointed out that this generic conversion from CLE to CBE has a critical flaw in the security proof. In 2012, Wu et al. [9] proposed a new generic conversion from CLE to CBE which involves the additional tool of collision resistant hash function. In 2009, Wu et. al proposed a general conversion from CLS to CBS, where the cryptographic hash function is involved and taken as a random oracle in the security proof. In this paper, we will deal with the generic conversion from CLS to CBS.

1.3. Contribtution

We propose a novel security model for CLS which will play an important role in the security proof for our generic conversion from CLS to CBS. In particular, our model allows the Type II attacker to attack the replaced public key, under the only condition that the attacker does not know the secret key for this current public key. For example, suppose that the PKG (Type II adversary) replaces the public key of the attacked user Alice with that of user Bob. In this way, the PKG does not know current private key, although he has already replaced public key. Hence, it is reasonable that the PKG remains unable to generate signature or decrypt ciphertexts under this replaced public key. In other words, if he can decrypt ciphertext or generate signatures in this way, this attack is interesting and should be taken into account. Additionally, we also redefine the signing oracle by considering the special case, i.e that the attacked public key is other user's public key (in this case, the private key is not known by the adversary). This is just the basic idea of our new security model. This new security property is usually ignored in previous definitions [4], but inherent in most existing secure CLS scheme, because our new security model still meets the basic rule that the attacker does not know the private key. This new security model will lead us to complete the security proof for our new conversion.

Next, we prove that any CLS scheme secure in this new security model can be directly transformed into a secure CBS scheme through our conversion method. It is obvious that the security proof does not needs any additional assumptions such as random oracles or collision resistant hash functions . In other words, any CLS scheme secure in the stand model can generate a CBS scheme secure in the standard model through our conversion. Additionally, to show the application of our result, a concrete provably secure CBS scheme is proposed.

1.4. Orgnization

ORGANIZATION. This paper is organized as follows. Section 2 reviews the syntax definition of CLS and proposes the new security model for CLS. Section 3 briefly reviews the definition and security model for CBS. Section 4 presents our new generic construction of CBS for CLS and its security proof. Section 5 presents a concrete CBS signature scheme as an application example of our generic conversion. At last, Section 6 draws the conclusion.

2. Certificateless Signauture

In this section, we first review the syntax definition of CLS [4], and then propose the new security model for CLS. At last, we compare our new security model with other existing ones. The new security definition formalizes new important property which has been ignored by other existing security models. In fact, it will play the key role in our new generic construction of CBS for CLS with provable security, and greatly helps to formally capture the intuitively close relation between CBS and CLS.

For writing convenience, we use the notation ID and ID to denote the identity information in the certificateless system and certificate based system respectively. We put the prefix $CL.$ to specify that this is in the certificateless system and $CB.$ specify that this is in the certificate based system. We use the notation $query \nrightarrow O$ to denote that the query $query$ has never been submitted to the oracle O, and the notation $answer \leftarrow O$ to denote that $answer$ has been returned by the oracle O.

2.1. Syntax of Certificateless Signature Scheme

Although there are some different syntax frameworks for certificateless signature schemes [4], there is very little essential difference between them. Here we directly adopt the CLS syntax provided by Huang et al. [4].

Definition 1. [Syntax of CLS]. A *certificateless signature scheme* consists of the following six algorithms.

- $\mathsf{CL.Setup}(1^k) \rightarrow (CL.msk, CL.param)$.

  It takes 1^k as input where k is the security parameter, and returns the master secret key $CL.msk$ and the parameter $CL.param$ which is shared in the system.

- $\mathsf{CL.ExtractPPK}(CL.msk, CL.param, \mathsf{ID}) \rightarrow D_{\mathsf{ID}}$.

  It takes the master private key msk, the system parameter $CL.param$ and the identity ID as input, and returns the partial private key D_{ID}.

- $\mathsf{CL.SetSV}(CL.param) \rightarrow X_{\mathsf{ID}}$.

  It takes as input the system parameter $CL.param$, and outputs a secret value X_{ID}.

- $\mathsf{CL.SetPK}(CL.param, X_{\mathsf{ID}}) \to (CL.PK_{\mathsf{ID}})$.

  It takes as input the system parameter $CL.param$, and this identity's secret value X_{ID}, and outputs the public key $CL.PK_{\mathsf{ID}}$.

- $\mathsf{CL.Sign}\ (CL.param, D_{\mathsf{ID}}, X_{\mathsf{ID}}, m) \to CL.\sigma$.

  It takes as input the system parameter $CL.param$, the partial private key D_{ID}, the secret value X_{ID} and the message m, and outputs the signature $CL.\sigma$.

- $\mathsf{CL.Verify}(CL.param, \mathsf{ID}, CL.PK_{\mathsf{ID}}, m, CL.\sigma) \to b \in \{0, 1\}$.

  It takes as input the system parameter $CL.param$, an identity ID, this identity's public key $CL.PK_{\mathsf{ID}}$ and a message/signature pair $(m, CL.\sigma)$, and outputs 1 if the signature is correct, or 0 otherwise.

2.2. Security Definition of CLS

In the following definition, if the adversary $\mathcal{A}$ is used to model the outside attacker who does not know the master secret key, it is said to be of Type I. If the adversary $\mathcal{A}$ is used to model the malicious authority who holds the master secret key, it is said to be of Type II. Both types of adversaries are allowed to mount the public key replacement attack, but they do this under different restrictions. For example, if the Type II adversary replaces the public key with the value generated by himself, he will know both the corresponding secret key and the partial private key. In this way, it can trivially generate signatures. Hence, in the definition, some restrictions should be made to avoid Type II adversaries replacing public key in this way. However, under the condition that the partial private key is not known by one Type I adversary, it can replace public keys in any way.

The signing oracles is divided two kinds: normal one or super one. The normal signing oracle $O^{CL.NSign}$ answers the signature query, only if he knows the corresponding secret value. The super signing oracle $O^{CL.NSign}$ is still forced to answer the signature query in some "super" way, even if it does not know the corresponding secret value. Generally speaking, both the secret value and the partial private key are the two indispensable factors for generating a CLS signature. Hence, for the challenger to answer the super signing oracle, some "super' method (such as depending random oracles) will be needed.

According to whether the signing oracle is normal or super and whether the type is I or II, the adversaries will be divided four kinds. In the following definition, against four different kinds of chosen message and chosen identity adversaries (CMCI), the existential unforgeability (EU) of CLS is defined in the uniform framework.

The following definition of ours is different from other ones including that in [4]. We will discuss the differences in the next subsection.

Definition 2 [Security of CLS]. A CLS scheme is CL-EUF-CMCI secure against a certain kind of adversary

$$\mathcal{A} \in \{ \text{CL.Normal-}\mathcal{A}^{I}, \text{CL.Super-}\mathcal{A}^{I}, \text{CL.Normal-}\mathcal{A}^{II}, \text{CL.Super-}\mathcal{A}^{II}\},$$

if no polynomially bounded adversary $\mathcal{A}$ has a non-negligible success probability in the following CLS game. In this following,

$$\text{CL.Normal-}\mathcal{A}^{I}, \text{CL.Super-}\mathcal{A}^{I}, \text{CL.Normal-}\mathcal{A}^{II}, \text{CL.Super-}\mathcal{A}^{II}$$

will be called normal Type I adversary, super Type I adversary, normal Type II adversary, and super Type II adversary respectively.
(1) Initial: The challenger runs the algorithm $\mathsf{CL.Setup}$, returns $CL.Params$ and the auxiliary information $aux \in \{nil, CL.msk\}$ to the attack $\mathcal{A}$ (nil means nothing), where

$$\begin{array}{ll} aux = nil, & \text{for } \mathcal{A} \in \{\text{CL.Normal-}\mathcal{A}^{I}, \text{ CL.Super-}\mathcal{A}^{I} \}; \\ aux = CL.msk, & \text{for } \mathcal{A} \in \{\text{CL.Normal-}\mathcal{A}^{II}, \text{CL.Super-}\mathcal{A}^{II}\}. \end{array}$$

(2) Queries: In this phase, $\mathcal{A}$ can adaptively make requests to a few oracles among the following ones.

- $O^{CL.CreateU}(\mathsf{ID}) \rightarrow CL.PK_{\mathsf{ID}}$
  This oracle receives an input ID and outputs this original public key of the identity ID.

- $O^{CL.ReplacePK}(\mathsf{ID}, CL.PK) \rightarrow \emptyset$
  For a public key replacement query $(\mathsf{ID}, CL.PK)$, it sets $CL.PK$ as the current public key.

- $O^{CL.SecretV}(CL.PK) \rightarrow X$
  If $CL.PK$ is the original public key of a ceratin identity ID (i.e., $CL.PK$ has been returned from the oracle $O^{CL.CreateU}$), this oracle returns the secret value X corresponding to $CL.PK$. Otherwise, it refuses this query.

- $O^{CL.PartialPK}(\mathsf{ID}, CL.PK) \to D_{\mathsf{ID}}$
  For a partial private key query ID, this oracle runs the algorithm $\mathsf{CL.PartialPK}$ and outputs the result D_{ID}.

- $O^{CL.NSign}(\mathsf{ID}, m) \to CL.\sigma$
  If

$$CL.\overline{PK}_{\mathsf{ID}} \leftarrow O^{CL.CreateU},$$

  which means that $CL.\overline{PK}_{\mathsf{ID}}$ has been provided by the oracle $O^{CL.CreateU}$, it outputs a valid signature $CL.\sigma$ of m under the current public key $CL.\overline{PK}_{\mathsf{ID}}$ of the identity ID. Otherwise, it refuses the query. Here note that more detailed discussion on this normal signing oracle will be presented in the next subsection.

- $O^{CL.SSign}(\mathsf{ID}, m) \to CL.\sigma$
  For a super signing query (ID, m), it outputs a valid signature $CL.\sigma$ of m under the current public key $CL.\overline{PK}_{\mathsf{ID}}$ of the identity ID.

Every attack model $\mathcal{A}$ has its own set $\mathcal{O}_{\mathcal{A}}$ of allowed oracles. In particular, with $\mathcal{O}' = \{O^{CL.CreateUser}, O^{CL.ReplacePK}, O^{CL.SecretV}\}$ being commonly allowed,

$$\begin{array}{ll}
\mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CL.NSign}, O^{CL.PartialPK}\}, & \text{for } \mathcal{A} = \text{CL.Normal-}\mathcal{A}^{I}; \\
\mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CL.SSign}, O^{CL.PartialPK}\}, & \text{for } \mathcal{A} = \text{CL.Super-}\mathcal{A}^{I}; \\
\mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CL.NSign}\}, & \text{for } \mathcal{A} = \text{CL.Normal-}\mathcal{A}^{II}; \\
\mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CL.SSign}\}, & \text{for } \mathcal{A} = \text{CL.Super-}\mathcal{A}^{II}.
\end{array}$$

(3) Output: After all queries, $\mathcal{A}$ outputs a forgery $(\mathsf{ID}^*, m^*, CL.\sigma^*)$. Let $CL.\overline{PK}_{\mathsf{ID}^*}$ be the current public key of ID^*. $\mathcal{A}$ is said to win the game if the forgery satisfies the following requirements. Here note that these restrictions are put forward to ensure that the forged signature is valid and this forgery is nontrivial. It is known that the basic requirement for CLS is that both the secret value and the partial private key are the two indispensable factors for generating a CLS signature. In other words, any forgery generated by the attacker who knows at most one of these two indispensable factors should be accepted as successful.

- For $\mathcal{A} \in$ {CL.Normal-$\mathcal{A}^{I}$, CL.Super-$\mathcal{A}^{I}$, CL.Normal-$\mathcal{A}^{II}$, CL.Super-$\mathcal{A}^{II}$}, it is commonly required that

$$1 = \mathsf{CL.Verify}(CL.mpk, CL.param, \mathsf{ID}^*, CL.\overline{PK}_{\mathsf{ID}^*}, m^*, CL.\sigma^*),$$

and

$$(\mathsf{ID}^*, m^*) \nrightarrow O^{CL.Sign},$$

where

$$\begin{array}{ll} O^{CL.Sign} = O^{CL.NSign}, & \text{for } \mathcal{A} \in \{\text{CL.Normal-}\mathcal{A}^I, \text{ CL.Normal-}\mathcal{A}^{II}\ \}; \\ O^{CL.Sign} = O^{CL.SSign}, & \text{for } \mathcal{A} \in \{\text{CL.Super-}\mathcal{A}^I, \text{CL.Super-}\mathcal{A}^{II}\}. \end{array}$$

Here, we use the notation $\nrightarrow$ to denote that the query (ID^*, m^*) has never been provided to the oracle O^{Sign}. This restriction ensures that the signature is valid and not trivially obtained from the signing oracle.

- For $\mathcal{A} \in \{\text{CL.Normal-}\mathcal{A}^I,\ \ \text{CL.Super-}\mathcal{A}^I\ \}$, additionally, it is required that

$$\mathsf{ID}^* \nrightarrow O^{CL.PartialPK}.$$

  This restriction ensures that at least the target partial private key is not known by $\mathcal{A}$.

- For $\mathcal{A} \in \{\text{CL.Normal-}\mathcal{A}^{II}, \text{ CL.Super-}\mathcal{A}^{II}\}$, additionally, it is required that

$$CL.\overline{PK}_{\mathsf{ID}^*} \nrightarrow O^{CL.SecretV},$$

  and

$$CL.\overline{PK}_{\mathsf{ID}^*} \leftarrow O^{CL.CreateU},$$

  where $CL.\overline{PK}_{\mathsf{ID}^*} \leftarrow O^{CL.CreateU}$ means that $CL.\overline{PK}_{\mathsf{ID}^*}$ is provided by the oracle $O^{CL.CreateU}$. This restriction ensures that the target secret value is not known by $\mathcal{A}$. This restriction is different to that in other security models such as that in [4]. This difference will be discussed in more details in the next subsection.

2.3. Comparison between Our Security Models and Others

In this section, we compare our security definition with other ones including that of [4]. By this comparison, it will be seen that our definition are more general, more reasonable and also be very useful.

Firstly, Huang et al. consider three kinds signing oracles, i.e., normal signing oracle, strong signing oracle, and super signing oracle, while we does not involve the strong signing oracle. Because the strong signing oracle is an intermediate one between the other ones and seldom mentioned in literature, we don't consider this signing oracle. However, with similar techniques to deal with the normal signing oracle and the super signing oracles, it is easy to deal with the strong oracles on the strong signing oracle. For more details, refer to [4].

Secondly, the normal signing oracle and the super signing oracle are different from others. This difference will play an important role in our generic construction of CBS schemes from CLS schemes. In [4], the normal signing oracle returns signatures, only if the current public key of the queried identity is the original one and this original public should belong to the queried identity. In contrast, in our definition, the normal signing oracle returns signatures, if the current public key is the original public key of any identity, i.e. not only the queried identity, but also other identities. Now consider an example case: the adversary obtains Alice's public key $CL.PK_{Alice}$ and Bob's public key $CL.PK_{Bob}$ from the oracle $O^{CL.CreateU}$, and then changes the public key of Alice into the value $CL.PK_{Bob}$ by making the oracle query $O^{CL.RepalcePK}(CL.ID_{Alice}, CL.PK_{Bob})$. In this case, the challenger knows the current secret value $X_{Alice} = X_{Bob}$ and can naturally answer the signing oracle under this replace public key. The signature query under this replaced public key is taken as normal in our definition. In contrast, in previous definitions [4], this signature query is taken as super. Our definition is more reasonable and will make senses in some cases. In fact, the initial reason to distinguish normal and super signing oracles is that: for the normal signing oracle, the challenger knows the master secret key and the secret value, and hence can naturally answer the singing oracles, while for the super signing oracle, the challenger should be "super" in the sense that he is required to answer the signing oracle without knowing the secret value.

Thirdly, the restriction for the forgery from Type II adversary, i.e.

$$CL.\overline{PK}_{\mathsf{ID}^*} \nrightarrow O^{CL.SecretV} \text{ and } CL.\overline{PK}_{\mathsf{ID}^*} \leftarrow O^{CL.CreateU},$$

may be more reasonable. This property will also play an important role in the generic construction of CBS schemes from CLS schemes. In [4], the target public key, as the original public key, is required to be the original public key of the target identity. In contrast, in our definition, the target public key can be any identity's original public key, only if it is "original" which can ensures that its secret value is not known by this Type II adversary. In fact, the initial reason for this restriction is to prevent the trivial case that: once the Type II adversary, who holds the master secret key, also knows the secret value, it can trivially forge signatures. In other words, any nontrivial forgery of the Type II adversary should be considered. Hence, this difference between our security definition and previous ones makes our definition involve the nontrivial forgery ignored in other security definitions. As we will see in next sections, it is this difference that matters whether we can get more intuitive, more simple, more efficient, provably secure generic construction from CLS to CBS.

At last, it is intuitive to see that many existing CLS signature schemes secure in the previous model will remain secure in our new model. In other words, what we has done is just to find out and formalize the new security property which has been holden by existing CLS signature schemes, but was ignored or not comprehensively captured in previous definitions. As you will see, our new general conversion method from CLS to CBS is more intuitive and efficient than that proposed by Wu et al. [8]. In particular, our conversion method does not depend on involving additional cryptographic tool, namely one hash function, which has to be taken as random oracle for the security proof to go through. At first, we can not provide the formal security proof for our new generic construction, although it is so simple, intuitive, and beautiful. When trying our best to find the reasons, we find some security properties of CLS which are indispensable for security proof but were never mentioned in previous security definition. This is the beginning of the procedure of revisiting the security model of CLS.

3. Certificate Based Signature

In this section, we directly adopt the definition and security model for CBS from [8]. Of course, we can improve the definition model in [8], as we did for CLS in the above section. In fact, it is not a difficult task. We aim to show that, from any CLS signature scheme, which is secure in our new security model, we can generally constructed a CBS signature which is secure in the previous security model [8]. Hence, we need to adopts and review the previous CBS signature definition.

Here, for the security definition of CBS, as we did for CLS, four kinds of adversaries are considered. In other words, we don't consider the strong signing oracle between the normal one and the super one. Additionally, the security definition is organized in a more compact framework. However, there is no essential difference between the following definition and that in [8].

Definition 3 [Syntax of CBS]. A Certificate Based Signature Scheme (CBS) consists of five algorithms as follows.

- $\mathsf{CB.Setup}(1^k) \to (CB.msk, CB.param)$.

  It takes as input the security parameter 1^k and returns the certifier's master secret key $CB.msk$ and the system parameter $CB.param$ that includes the description of a string space Γ, which can be any subset of $\{0,1\}^*$.

- $\mathsf{CB.GenUK}(CB.param) \to (CB.PK_{ID}, CB.SK_{ID})$.

  It takes input the system parameter $CB.param$, and outputs the secet/public key pair (SK_{ID}, PK_{ID}) for a certain entity ID.

- $\mathsf{CB.Cert}(CB.msk, CB.param, ID, CB.PK_{ID}) \to cert_{ID}$.

  It takes as input the master secret key $CB.msk$, the system parameter $CB.param$, the identity ID and its public key $CB.PK_{ID}$, and outputs the certificate $cert_{ID}$.

- $\mathsf{CB.Sign}(CB.param, ID, CB.PK_{ID}, cert_{ID}, CB.SK_{ID}, m) \to CB.\sigma$.

  It takes as input the system parameter $CB.param$, the identity ID, the public key $CB.PK$, the certificate $cert_{ID}$, the secret key $CB.SK_{ID}$ and the message m, and outputs the signature $CB.\sigma$.

- $\mathsf{CB.Verify}(CB.param, ID, CB.PK_{ID}, m, CB.\sigma) \to b \in \{0,1\}$.

  It takes as input the system parameter $CB.param$, an identity ID, this identity's public key $CB.PK_{ID}$ and a message/signature pair $(m, CB.\sigma)$, and outputs 1 if the signature is correct, or 0 otherwise.

Definition 2 [Security of CBS]. A CBS scheme is CB-EUF-CMCI secure against a certain kind of adversary

$$\mathcal{A} \in \{\text{ CB.Normal-}\mathcal{A}^I\text{,CB.Super-}\mathcal{A}^I\text{,CB.Normal-}\mathcal{A}^{II}\text{,CB.Super-}\mathcal{A}^{II}\},$$

if no polynomially bounded adversary $\mathcal{A}$ has a non-negligible success probability in the following CBS game. In this following,

$$\text{CB.Normal-}\mathcal{A}^I\text{,CB.Super-}\mathcal{A}^I\text{,CB.Normal-}\mathcal{A}^{II}\text{,CB.Super-}\mathcal{A}^{II}$$

will be called normal Type I adversary, super Type I adversary, normal Type II adversary, and super Type II adversary respectively.
(1) Initial: The challenger runs the algorithm CB.Setup, returns $CB.Params$ and the auxiliary information $aux \in \{nil, CB.msk\}$ to the attack $\mathcal{A}$ (nil means nothing), where

$$\begin{array}{ll} aux = nil, & \text{for } \mathcal{A} \in \{\text{CB.Normal-}\mathcal{A}^I\text{, CB.Super-}\mathcal{A}^I\ \}; \\ aux = CB.msk, & \text{for } \mathcal{A} \in \{\text{CB.Normal-}\mathcal{A}^{II}\text{,CB.Super-}\mathcal{A}^{II}\}. \end{array}$$

(2) Queries: In this phase, $\mathcal{A}$ can adaptively make requests to a few oracles among the following ones.

- $O^{CB.CreateU}(ID) \to CB.PK_{ID}$
  This oracle receives an input ID and outputs this original public key of the identity ID.

- $O^{CB.ReplacePK}(ID, CB.PK) \to \emptyset$
  For a public key replacement query $(ID, CB.PK)$, it sets $CB.PK$ as the current public key of ID.

- $O^{CB.Corrupt}(ID) \to CB.SK_{ID}$
  If ID has been submitted to the oracle $O^{CB.CreateU}$, this oracle returns the secret key $CB.SK_{ID}$ corresponding to ID's original public key. Otherwise, it first makes the oracle query $O^{CB.CreateU}(ID)$ and then the oracle query $O^{CB.Corrupt}(ID)$.

- $O^{CB.Cert}(ID, CB.PK) \to cert_{ID}$
  For this certification query, this oracle gets the result $cert_{ID}$ by running the algorithm CB.Cert and outputs it.

- $O^{CB.NSign}(ID, m) \to CB.\sigma$
  If ID has been submitted to $O^{CB.CreateU}$, it outputs a valid signature $CB.\sigma$ of m under the original public key $CB.PK_{ID}$ of the identity ID. Otherwise, it refuses the query.

- $O^{CB.SSign}(ID, m) \rightarrow CB.\sigma$
  For a super signing query (ID, m), it outputs a valid signature $CB.\sigma$ of m under the current public key $CB.\overline{PK}_{ID}$ of the identity ID.

Every attack model $\mathcal{A}$ has its own set $\mathcal{O}_{\mathcal{A}}$ of allowed oracles. In particular, with $\mathcal{O}' = \{O^{CB.CreateUser}, O^{CB.ReplacePK}, O^{CB.Corrupt}\}$ being commonly allowed,

$$\begin{array}{ll} \mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CB.NSign}, O^{CB.Cert}\}, & \text{for } \mathcal{A} = \text{CB.Normal-}\mathcal{A}^{I}; \\ \mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CB.SSign}, O^{CB.Cert}\}, & \text{for } \mathcal{A} = \text{CB.Super-}\mathcal{A}^{I}; \\ \mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CB.NSign}\}, & \text{for } \mathcal{A} = \text{CB.Normal-}\mathcal{A}^{II}; \\ \mathcal{O}_{\mathcal{A}} = \mathcal{O}' \cup \{O^{CB.SSign}\}, & \text{for } \mathcal{A} = \text{CB.Super-}\mathcal{A}^{II}. \end{array}$$

(3) Output: After all queries, $\mathcal{A}$ outputs a forgery $(ID^*, m^*, CB.\sigma^*)$. Let $CB.\overline{PK}_{ID^*}$ be the current public key of ID^*. $\mathcal{A}$ is said to win the game if the forgery satisfies the following requirements.

- For $\mathcal{A} \in$ {CB.Normal-$\mathcal{A}^{I}$, CB.Super-$\mathcal{A}^{I}$, CB.Normal-$\mathcal{A}^{II}$, CB.Super-$\mathcal{A}^{II}$}, it is commonly required that

$$1 = \mathsf{CB.Verify}(CB.mpk, CB.param, ID^*, CB.\overline{PK}_{ID^*}, m^*, CB.\sigma^*),$$

  and

$$(ID^*, m^*) \nrightarrow O^{Sign},$$

  where

$$\begin{array}{ll} O^{CB.Sign} = O^{CB.NSign}, & \text{for } \mathcal{A} \in \{\text{CB.Normal-}\mathcal{A}^{I}, \text{ CB.Normal-}\mathcal{A}^{II}\}; \\ O^{CB.Sign} = O^{CB.SSign}, & \text{for } \mathcal{A} \in \{\text{CB.Super-}\mathcal{A}^{I}, \text{CB.Super-}\mathcal{A}^{II}\}. \end{array}$$

- For $\mathcal{A} \in$ {CB.Normal-$\mathcal{A}^{I}$, CB.Super-$\mathcal{A}^{I}$ }, additionally, it is required that

$$ID^* \nrightarrow O^{CB.Cert}.$$

- For $\mathcal{A} \in$ {CB.Normal-$\mathcal{A}^{II}$, CB.Super-$\mathcal{A}^{II}$}, additionally, it is required that

$$ID^* \nrightarrow O^{CB.Corrupt},$$

  and

$$CB.\overline{PK}_{ID^*} = O^{CB.CreateU}(ID^*).$$

4. Generic Construction CLS-2-CBS and Security Proof

In this section, we will introduce a generic method to construct certificate based signatures from certificateless signatures. We show that this CBS signature generated in this way is provably secure against any kind of adversary such as CB.Normal-$\mathcal{A}_I$, only if the underlying CLS signature scheme is provably secure against the corresponding kind of adversary CL.Normal-$\mathcal{A}_I$. At last we compare our conversion method from CLS to CBS with that in [8].

4.1. Generic Construction CLS-2-CBS

Let Π^{CL} be a CLS scheme with algorithms:

$$\Pi^{CL} = (\mathsf{CL.Setup}, \mathsf{CL.SetSV}, \mathsf{CL.SetPK}, \mathsf{CL.ExtractPPK}, \mathsf{CL.Sign}, \mathsf{CL.Verify}),$$

as specified in Definition 1. Then a CBS scheme

$$\Pi^{CL} = (\mathsf{CB.Setup}, \mathsf{CB.GenUK}, \mathsf{CB.Cert}, \mathsf{CB.Sign}, \mathsf{CB.Verify})$$

is defined as follows. Let Γ be the identity information space for Π^{CB}, $\mathcal{PKCB}$ be the public key space for Π^{CB} and $\mathcal{IDCL}$ denotes the space of identities for Π^{CL}. Without loss of generality, we assume that $\mathcal{IDCL} = \Gamma \times \mathcal{PKCB}$.

- $\mathsf{CB.Setup}$.
  On input a security parameter k, first run

  $$(CL.msk, CL.param) \leftarrow \mathsf{CL.Setup}(1^k).$$

  Then set $CB.msk = CL.msk$. Define $CB.param$ by extending $CL.param$ to include a description of Γ which is the identity information space for Π^{CB}. The output is $(CB.msk, CB.param)$.

- $\mathsf{CB.GenUK}$.
  On input $CB.param$, first extract $CL.param$ from $CB.param$. Run

  $$\begin{aligned} X &\leftarrow \mathsf{CL.SetSV}(CL.param), \\ CL.PK &\leftarrow \mathsf{CL.SetPK}(CL.param, X). \end{aligned}$$

  The output is $(CB.PK, CB.SK) = (CL.PK, X)$.

- CB.Cert.
  On input $CB.msk, CB.param, ID, CB.PK_{ID}$, first extract $CL.param$ from $CB.param$. Set the $\mathsf{ID} = ID\,||CB.PK_{ID}$ and $CL.msk = CB.msk$. The output is

$$cert_{ID} = \mathsf{CL.ExtractPPK}\ (CL.param,\ CL.msk, \mathsf{ID}).$$

- CB.Sign.
  On input $CB.param,\ cert_{ID},\ \ CB.SK_{ID},\ m$, first extract $CL.param$ from $CB.param$. Then set $\mathsf{ID} = ID||CB.PK_{ID}$, $CL.PK_{\mathsf{ID}} = CB.PK_{ID}$, $D_{\mathsf{ID}} = cert_{ID}$, $X_{\mathsf{ID}} = CB.SK_{ID}$. The output is

$$CB.\sigma = \mathsf{CL.Sign}(CL.param,\ \mathsf{ID},\ CL.PK_{\mathsf{ID}},\ D_{\mathsf{ID}},\ X_{\mathsf{ID}}, m).$$

- CB.Verify.
  On input $CB.param, ID, CB.PK_{ID}, m, CB.\sigma$, extract $CL.param$ from $CB.param$. Set $\mathsf{ID} = ID||CB.PK_{ID}$, $CL.PK_{ID} = CB.PK_{ID}$ and $CL.\sigma = CB.\sigma$. The output is

$$b = \mathsf{CL.Verify}(CL.param, \mathsf{ID}, CL.PK_{\mathsf{ID}}, m,\ CL.\sigma).$$

4.2. Security Proof for CLS-2-CBS

In the following, we prove four theorems which deal with the four kinds of adversaries respectively. In particular, if the CLS scheme is secure against one certain kind of adversary in our new security model, then the corresponding CBS signature is also secure against the corresponding kind of adversary in the security model reviewed in section 3.

Theorem 1. Suppose that $\mathcal{A}^I$ is a super Type I adversary against Π^{CB} with success probability ϵ and running time t. Then there is a super Type I adversary $\mathcal{B}^I$ against Π^{CL} with success probability ϵ and running time $O(t)$.

Proof. Let $\mathcal{C}$ denote the Π^{CL} challenger against $\mathcal{B}^I$. $\mathcal{B}^I$ mounts a Type I attack on Π^{CL} by simulating the challenger for $\mathcal{A}^I$ and using help from $\mathcal{A}^I$ as follows.

Initial phase for CBS game. $\mathcal{B}^I$ obtains from $\mathcal{C}$ the system parameter of Π^{CL} and extends it into the system parameter $CB.param$ of Π^{CB} as done in CB.Setup of Π^{CB}. $\mathcal{B}^I$ supplies $CB.param$ to $\mathcal{A}^I$.

Queries Phase for CBS game. When $\mathcal{A}^I$ enters the Queries phase for the CBS game, $\mathcal{B}^I$ accordingly enters the Queries phase of the CLS game. For the oracle queries from $\mathcal{A}^I$, $\mathcal{B}^I$ handles these queries as follows.

- $O^{CB.CreateU}(ID) \to CB.PK_{ID}$.
  If the query ID has been submitted to the oracle $O^{CB.CreateU}$, it will directly returns the previous answer which is recorded in the list L. Otherwise, it does as follows. $\mathcal{B}^I$ chooses a random identity ID' and obtains its original public key $CL.\widetilde{PK}_{\mathsf{ID}'}$ through the oracle $O^{CL.CreateU}$. It sets $\mathsf{ID} = ID\ ||CB.PK_{\mathsf{ID}'}$ and requires the oracle $O^{CL.ReplacePK}$ to change the public key of ID into $CL.\widetilde{PK}_{\mathsf{ID}'}$. In this case, the CBS original public key of ID is the CLS original public key of ID'. In other words, every original public key for Π^{CB} is related with a certain original public key for Π^{CL}.

  Additionally, to record the above operations for future use, $\mathcal{B}^I$ sets the original public key
  $$CB.\widetilde{PK}_{ID} = CL.\widetilde{PK}_{\mathsf{ID}'}$$
  and the current public key
  $$CB.\overline{PK}_{ID} = CL.\widetilde{PK}_{\mathsf{ID}'},$$
  and adds the tuple
  $$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID})$$
  to the initially empty list L.

- $O^{CB.ReplacePK}(ID, CB.PK) \to \emptyset$
  If ID has not been submitted to the oracle $O^{CB.CreateU}$, $\mathcal{B}^I$ first makes the query $O^{CB.CreateU}(ID)$ by himself before doe the following. Otherwise, it directly does the following. $\mathcal{B}^I$ sets $\mathsf{ID} = ID\ ||CB.PK$, and sequentially makes two oracle queries $O^{CL.CreateU}(\mathsf{ID})$ and $O^{CL.ReplacePK}$ $(\mathsf{ID}\ , CB.PK)$ to change the public key value of ID into $CB.PK$.

  Additionally, to record the above operation, $\mathcal{B}^I$ searches the relative tuple
  $$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID}),$$
  and then changes the value of $CB.\overline{PK}_{ID}$ into $CB.PK$.

- $O^{CB.Corrupt}(ID) \to CB.SK_{ID}$
  Without loss of generality, we assume that ID has been submitted to the oracle $O^{CB.CreateU}$. $\mathcal{B}^I$ searches the corresponding tuple
  $$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID})$$

in the list L. Then it makes the oracle query $O^{CL.SecretV}(CB.\widetilde{PK}_{ID})$ and relays this answer to the attacker $\mathcal{A}^I$. Here note $CB.\widetilde{PK}_{ID} = CL.\widetilde{PK}_{\mathsf{ID}'}$.

- $O^{CB.Cert}(ID, CB.PK) \rightarrow cert_{ID}$
  Without loss of generality, we assume that the current public key $CB.\overline{PK}_{ID} = CB.PK$ in the corresponding tuple
  $$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID}).$$
  of the list L. $\mathcal{B}^I$ sets $\mathsf{ID} = ID\ ||CB.PK$, makes the oracle query $O^{CL.PartialPK}(\mathsf{ID})$, and then relays the returned partial private key D_{ID} as the certificate for $\mathcal{A}^I$.

- $O^{CB.SSign}(ID, m) \rightarrow CB.\sigma$
  For a query (ID, m), this oracle browses the list L for the corresponding tuple
  $$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID}).$$
  Then it sets $\mathsf{ID} = ID||CB.\overline{PK}_{ID}$ and makes the signing oracle query $O^{CL.SSign}(\mathsf{ID}, m)$ and relays this answer to $\mathcal{A}^I$.

Output for CBS game. Now the attacker $\mathcal{A}^I$ returns its forgery (ID^*, m^*, $CB.\sigma^*$). Without loss of generality, we assume that $\mathcal{A}^I$ has made the oracle query $O^{CB.CreateU}$ or the replacing public key oracle query for ID^*. $\mathcal{B}^I$ browses the list L for the the corresponding tuple

$$(ID^*, \mathsf{ID}'^*, CB.\widetilde{PK}_{ID^*}, CB.\overline{PK}_{ID^*}),$$

and returns $(\mathsf{ID}^*, m^*, CL.\sigma^*)$ to its challenger $\mathcal{C}$, where

$$\mathsf{ID}^* = ID^*||CB.\overline{PK}_{ID^*}, CL.\sigma^* = CB.\sigma^*.$$

Analysis. First, from the relations of Π^{CB} and Π^{CL}, it can be easily or trivially seen that $\mathcal{B}^I$ perfectly simulates the game settings for $\mathcal{A}^I$ in the two phases of Initial and Queries. Second, if the forgery (ID^*, m^*, $CB.\sigma^*$) is successful, i.e., this forgery satisfies the three additions:

$$\begin{array}{rcl} \mathsf{CB.Verify}(CB.param, ID^*, CB.\overline{PK}_{ID^*}, m*, CB.\sigma^*) & = & 1, \\ (ID^*, m^*) & \nrightarrow & O^{CB.SSign}. \\ (ID^*, CB.\overline{PK}_{ID^*}) & \nrightarrow & O^{CB.Cert}. \end{array}$$

then, by checking these two groups of three restrictions one by one (the above and the below), it easily follows that $(\mathsf{ID}^*, m^*, CL.\sigma^*)$ is also successful, i.e. this forgery satisfies that

$$\begin{array}{rcl} \mathsf{CL.Verify}(CL.param, \mathsf{ID}^*, CL.\overline{PK}_{\mathsf{ID}^*}, m^*, CL.\sigma^*) & = & 1, \\ (\mathsf{ID}^*, m^*) & \nrightarrow & O^{CL.SSign}. \\ \mathsf{ID}^* & \nrightarrow & O^{CL.PartialPK}. \end{array}$$

Hence, the success probability of $\mathcal{B}^I$ is same to that of $\mathcal{A}^I$. Additionally, since what $\mathcal{B}^I$ mainly does in reduction is just issuing some relative queries to $\mathcal{C}$, it is obvious that the time of $\mathcal{B}^I$ is almost equal to the time t of $\mathcal{A}^I$. Hence we say that the running time of $\mathcal{B}^I$ is $O(t)$. □

Theorem 2. Suppose that $\mathcal{A}^{II}$ is a super Type II adversary against Π^{CB} with success probability ϵ and running time t. Then there is a super Type II adversary $\mathcal{B}^{II}$ against Π^{CL} with success probability ϵ and $O(t)$.

Proof. Let $\mathcal{C}$ denote a Π^{CL} challenger against Type II adversary $\mathcal{B}^{II}$. $\mathcal{B}^{II}$ mounts a Type II attack on Π^{CL} using help from $\mathcal{A}^{II}$ as follows.

Initial for CBS game. How $\mathcal{B}^{II}$ communicates with $\mathcal{A}^{II}$ is the same to how $\mathcal{B}^I$ communicates with $\mathcal{A}^I$ in the above proof for lemma 1, except that $\mathcal{B}^{II}$ additionally gets the master private key $CL.msk$ and relays it to $\mathcal{A}^{II}$ as the master key $CB.msk$ for Π^{CB}.

Queries for CBS game. How $\mathcal{B}^{II}$ answers these queries from $\mathcal{A}^{II}$ is the same to how $\mathcal{B}^I$ simulates the oracles for $\mathcal{A}^I$ in the above proof for lemma 1, except that $\mathcal{A}^{II}$ does not query the oracle $O^{CB.Cert}$.

Output for CBS game. How $\mathcal{B}^{II}$ generates the forged signature is the same to how $\mathcal{B}^I$ generates the forged signature in the above proof for lemma 1. Of course, the forged signatures from $\mathcal{B}^I$ and $\mathcal{B}^{II}$ satisfy different conditions.

Analysis. This analysis can immediately follows from the analysis in the above proof of lemma 1. Here, we only deal with the somewhat difficult part of the analysis. In particular, we will show how to get

$$CL.\overline{PK}_{\mathsf{ID}^*} \nrightarrow O^{CL.SecretV}, \text{ and } CL.\overline{PK}_{\mathsf{ID}^*} \leftarrow O^{CL.CreateU},$$

from

$$ID^* \nrightarrow O^{CB.Corrupt}, \text{ and } CB.\overline{PK}_{ID^*} = O^{CB.CreateU}(ID^*),$$

where

$$\mathsf{ID}^* = ID^* || CB.\overline{PK}_{ID^*}, CL.\sigma^* = CB.\sigma^*.$$

By checking the simulation of the oracle $O^{CB.CreateU}$ by $\mathcal{B}^{II}$ for $\mathcal{A}^{II}$ (same to those in the proof of Theorem 1), the equation $CB.\overline{PK}_{ID^*} = O^{CB.CreateU}(ID^*)$ means that the original public key $CB.\widetilde{PK}_{ID^*}$ is equal to the current public key $CB.\overline{PK}_{ID^*}$, i.e. that

$$CB.\widetilde{PK}_{ID^*}{=}CB.\overline{PK}_{ID^*}.$$

By checking the simulation of oracles $O^{CB.CreateU}$ and $O^{CB.ReplacePK}$, it can be seen that the CLS current public key $CL.\overline{PK}_{\mathsf{ID}}$ of $\mathsf{ID} = ID||CB.\overline{PK}_{ID}$ is always equal to the CBS current public key $CB.\overline{PK}_{ID}$ of ID. in particular, for $\mathsf{ID}^* = ID^*||CB.\overline{PK}_{ID^*}$, we have

$$CB.\overline{PK}_{ID^*} = CL.\overline{PK}_{\mathsf{ID}^*}.$$

Let the tuple $(ID^*, \mathsf{ID}'^*, CB.\widetilde{PK}_{ID^*}, CB.\overline{PK}_{ID^*})$ be the corresponding record in L. Then by checking the simulation of the oracle $O^{CB.CreateU}$ again, it can be seen that the CBS original public key $CB.\widetilde{PK}_{ID^*}$ is equal to the CLS original public key $CL.\widetilde{PK}_{\mathsf{ID}'^*}$ of ID'^*, i.e. that

$$CB.\widetilde{PK}_{ID^*} = CL.\widetilde{PK}_{\mathsf{ID}'^*}\ .$$

By the above three equations, we can get

$$CL.\overline{PK}_{\mathsf{ID}^*} = CL.\widetilde{PK}_{\mathsf{ID}'^*}\ .$$

Hence, from

$$CB.\overline{PK}_{ID^*} = O^{CB.CreateU}(ID^*),$$

we can get that

$$CL.\overline{PK}_{\mathsf{ID}^*} \leftarrow O^{CL.CreateU}.$$

By checking the oracles simulation process provided by $\mathcal{B}^{II}$ to $\mathcal{A}^{II}$ (same to those in the proof of Theorem 1), it can be seen that $O^{CL.SecretV}(CL.\widetilde{PK}_{\mathsf{ID}'^*})$ is queried by $\mathcal{B}^{II}$ only when $O^{CB.Corrupt}(ID^*)$ is queried by $\mathcal{A}^{II}$. Then by the just proved equation $CL.\overline{PK}_{\mathsf{ID}^*} = CL.\widetilde{PK}_{\mathsf{ID}'^*}$, we prove that $O^{CL.SecretV}$ $(CL.\overline{PK}_{\mathsf{ID}^*})$ is queried by $\mathcal{B}^{II}$ only when $O^{CB.Corrupt}(ID^*)$ is queried by $\mathcal{A}^{II}$. In other words, from

$$ID^* \nrightarrow O^{CB.Corrupt}$$

we can get

$$CL.\overline{PK}_{\mathsf{ID}^*} \nrightarrow O^{CL.SecretV}. \ \square$$

Theorem 3. Suppose that $\mathcal{A}^I$ is a normal Type I adversary against Π^{CB} with success probability ϵ and running time t. Then there is a normal Type I adversary $\mathcal{B}^I$ against Π^{CL} with success probability ϵ and running time $O(t)$.
Proof. The proof for Theorem 3 is the same to that for Theorem 1, except the difference that the CBS normal signing oracle is simulated depending on the CLS normal signing oracle in this proof for Theorem 3, while the CBS super signing oracle is simulated depending on the CLS super signing oracle in this proof for Theorem 1. Now show how to simulate the CBS normal signing oracle using the CLS normal signing oracle.

For a normal signing query $O^{CB.NSign}(ID, m)$, browse the list L for the corresponding tuple

$$(ID, \mathsf{ID}', CB.\widetilde{PK}_{ID}, CB.\overline{PK}_{ID}).$$

set $\mathsf{ID} = ID||CB.\overline{PK}_{ID}$ and make the signing oracle query $O^{CL.NSign}(\mathsf{ID}, m)$ and relays this answer to $\mathcal{A}^I$. By checking the simulation of the oracle $O^{CB.CreateU}$, it can be seen that the original CBS public key $CB.\widetilde{PK}_{ID}$ of ID is the original CLS public key $CL.\widetilde{PK}_{\mathsf{ID}'}$ of ID', and the original CLS public key $CL.\widetilde{PK}_{\mathsf{ID}'}$ of ID' is the current CLS public key $CL.\overline{PK}_{\mathsf{ID}}$ of ID. Hence, the query (ID, m) will be not rejected by the oracle $O^{CL.NSign}$. Hence, this proof can immediately follow Theorem 1. $\square$

Theorem 4. Suppose that $\mathcal{A}^I$ is a normal Type II adversary against Π^{CB} with success probability ϵ and running time t. Then there is a normal Type II adversary $\mathcal{B}^I$ against Π^{CL} with success probability ϵ and running time $O(t)$.
Proof. The proof for Theorem 4 is the same to that for Theorem 2, except the difference that the CBS normal signing oracle is simulated depending on the CLS normal signing oracle in this proof for Theorem 4, while the CBS super signing oracle is simulated depending on the CLS super signing oracle in this proof for Theorem 2. Additionally, in the proof of theorem 3, we have showed how to simulate the CBS normal signing oracle using the CLS normal signing oracle. Hence, this proof can immediately follow Theorem 2 and Theorem 3. $\square$

4.3. Comparison between CLS-2-CBS with Wu et al.'s Construction

5. Application Example - One Concrete CBS scheme

To demonstrate the application of the CLS-2-CBS conversion, this section describes a concrete CBS scheme from a CLS scheme proposed in [4]. We start by reviewing the bilinear groups and the related complexity assumption.

5.1. Bilinear Pairing and Complexity Assumption

This section briefly reviews the definition of bilinear pairings and the related complexity assumptions.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of prime order q and let P be a generator of $\mathbb{G}_1$, where $\mathbb{G}_1$ is additively represented and $\mathbb{G}_2$ is multiplicatively. A map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is said to be a bilinear pairing, if the following three conditions hold: (1)e is bilinear, i.e. $e(aP, bP) = e(P, P)^{ab}$ for all $a, b \in \mathbb{Z}_q^*$; (2) e is non-degenerate, i.e. $e(P, P) \neq 1$, where 1 is the identity of $\mathbb{G}_2$; (3) e is efficiently computable.

Definition 5 [Computational Diffie-Hellman Problem,CDH] Given P, aP, bP with uniformly random choices of $a, b \in \mathbb{Z}_q$, output abP. An algorithm $\mathcal{A}$ has success probability ϵ in solving the CDH problem, if

$$Pr[\mathcal{A}(P, aP, bP) = abP] = \epsilon].$$

The CDH problem is said to be (t, ϵ)-intractable if there is no algorithm to solve this problem with time less than t and success probability greater than ϵ.

5.2. Concrete CBS Scheme

The scheme described in this section is constructed based on the CLE scheme in [2] through the generic conversion $CLS-2-CBS$. It consists of following algorithms.

- $\mathsf{Setup}(1^k) \to (msk, params)$.

  This algorithm is run by the authority CA (key generating center). Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of prime order q, P be a generator of $\mathbb{G}_1$, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing. It specifies two hash functions $H_0, H_1 : \{0, 1\}^* \to \mathbb{G}_1^*$. Let Γ be the set of identity information. It chooses its master key $msk = s$ uniformly at random from $\mathbb{Z}_q$ and computes its public key $mpk = sP$. The system parameter

$$params = (q, e, \mathbb{G}_1, \mathbb{G}_2, P, mpk, H_0, H_1, \Gamma).$$

- $\mathsf{GenUK}(params) \to (PK_{ID}, SK_{ID})$.

  It selects a random $SK_{ID} \in Z_q$, computes $PK_{ID} = SK_{ID}P$ and outputs SK_{ID}, PK_{ID} as ID's secret/public key pair.

- $\mathsf{Cert}(CB.msk, params, ID, PK_{ID}) \to cert$.

  It sets $cert_{ID} = sH_0(ID||PK_{ID})$.

- $\mathsf{Sign}(params, ID, cert_{ID}, PK_{ID}, SK_{ID}, m) \to \sigma$. For a message m, the user ID computes the signature $\sigma = (u, v, W)$ where

  - $u = H_1(m||ID||PK_{ID}||r_1P||e(P,P)^{r_2})$ for random $r_1, r_2 \in Z_q$, which are chosen by ID.
  - $v = r_1 - u \cdot SK_{ID} \bmod q$, $W = r_2P - u \cdot cert_{ID}$.

- $\mathsf{Verify}(params, ID, PK_{ID}, m, \sigma) \to b$. Given a message/signature pair $(m, \sigma = (u, v, W))$, user ID's public key PK_{ID}, anyone can check whether

  $$u = H_1(m||ID||PK_{ID}||vP + u \cdot PK_{ID}||e(W,P)e(H_0(ID||PK_{ID}), sP)^u),$$

  If the equality holds, this algorithm outputs 1 . Otherwise, it outputs 0.

Theorem 5. The above CBS scheme is secure (in the random oracle model) against CB.Super-$\mathcal{A}^I$ and CB.Super-$\mathcal{A}^{II}$, assuming that CDH problem is hard in $\mathbb{G}_1$.

Proof. The proof follows Theorem 1, 2 and the fact that the underlying CLS scheme is secure against CL.Super-$\mathcal{A}^I$ and CL.Super-$\mathcal{A}^{II}$ if the CDH problem is hard in $\mathbb{G}_1$. In [4], the underlying CLS scheme is proved secure in their security model. As we have discussed, our security definition against super Type I adversary is same to that for the super Type I adversary. Hence, the CLS signature scheme is secure against the super Type I adversary in our model. Our security definition against super Type II adversary is same to that for the super Type II adversary of Huang et al. in [4], except that Huang et al.'s definition [4] requires the target public key key to be the original public key of the target identity, but our security only requires the secret key for the target public key is not known by the the Type II adversary. Because the difference between the definitions for super Type II adversary has very little effects on the security proof, the security proof for this CLS signature scheme security against super Type II adversary in our model can be immediately obtained from the proof of security against (super) Type II adversary in the model of [4]. In fact, in the proof of [4], what matters for the proof is not that the target public key is "original", but that the corresponding secret key is not known by this adversary. Here we omitted the trivial details. □

6. Conclusion

In this paper, we proposed a new provably secure generic conversion from CLS to CBS. It is very neat in the sense that there is no additional cryptographic tool involved for the conversion itself or its security proof. To analyze its security, we redefined the security model for CLS by formalizing some important but previously ignored property which is the main factor making our new conversion provably secure. As an example, we constructed a new provably secure certificate based signature scheme by applying this new generic method. In the theoretic sense, our results formally showed that the two conceptions of CBS and CLS (CBC and CLC) are very closely connected to each other.

References

[1] S. S.Al-Riyami, K. G.Paterson. Certificateless public key cryptography. in:Advancesin Cryptology-ASIACRYPT 2003, pp.452-473.

[2] S.S. Al-Riyami and K.G. Paterson. CBS from CLS: A generic construction and ecient schemes. Public Key Cryptography - PKC 2005, Lecture Notes in Comput. Sci., vol. 3386, pp. 398-415, 2005.

[3] C. Gentry. Certificate-based Signature and the certicate revocation problem. Advances in Cryptology - EUROCRYPT 2003, Lecture Notes in Comput. Sci., vol. 2656, pp. 272-293, 2003.

[4] X. Huang, Y. Mu, W. Susilo, D. Wong, W. Wu, "Certificateless Signatures: New Schemes and Security Models," The Computer Journal, Oxford, September 29, 2011 doi:10.1093/comjnl/bxr097

[5] B.G. Kang and J.H. Park. Is it possible to have CBS from CLS?. eprint.iacr.org/2005/431.ps

[6] B.G. Kang, J.H. Park, S.G. Hahn. A certificate-based signature scheme. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 99C111. Springer, Hei-delberg (2004)

[7] A. Shamir. Identity-based cryptosystems and signature schemes. Crypto 1984, LNCS Vol. 196, pp. 47-53, 1984.

[8] W Wu, Y. Mu, W. Susilo, X. Huang: Certificate-based Signatures Revisited. Journal of Universal Computer Science 15(8): 1659-1684 (2009)

[9] W. Wu, Y. Mu, W. Susilo, X. Huang and L. Xu. Provably Secure Construction of Certificate-based Encryption from Certificateless Encryption. The Computer Journal Advance Access published January 5, 2012, doi:10.1093/comjnl/bxr130.

[10] D.H. Yum and P.J. Lee. Generic Construction of Certificateless Signature. In Computational Science and Its Applications - ICCSA 2004 , LNCS vol. 3043, pp. 802-811, Springer-Verlag, 2004