Information Sciences 296 (2015) 315–321

Contents lists available at ScienceDirect

Information Sciences

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

Security models for certificateless signature schemes revisited

Kyung-Ah Shim

National Institute for Mathematical Sciences, KT Daedoek 2nd Research Center, 463-1 Jeonmin-dong, Yuseong-gu, Daejeon, Republic of Korea

ARTICLE INFO

Article history:
Received 21 May 2008
Received in revised form 5 June 2014
Accepted 24 October 2014
Available online 11 November 2014

Keywords:
Digital signature
Identity-based system
Certificateless signature
Key replacement attack
Formal security model

ABSTRACT

Certificateless cryptography eliminates the need of certificates in the Public Key Infrastructure and solves the inherent key escrow problem in the ID-based cryptography. In this paper, we point out security pitfalls on the restrictions of an adversary's final output in security models of certificateless signature schemes by demonstrating key replacement attacks on three certificateless signature schemes in the different security models.

1. Introduction

Public-key cryptographys (PKCs) need authentication of users' public keys. Public Key Infrastructure (PKI) is an arrangement that binds public keys with respective user identities by means of public key certificates issued by a certificate authority (CA). The public key certificate is an electronic document which contains the CA's signature to bind a public key with an identity information. The certificate can be used to verify that a public key belongs to an individual. This PKI causes several problems of certificate management including storage, distribution and the computational cost of certificate verification. Identity (ID)-based infrastructure [12] allows a user's public key to be easily derived from its known identity information by eliminating the need for public-key certificates. Such cryptosystems alleviate certificate overhead and solve the problems of PKI technology. In ID-based infrastructure, a Private Key Generator (PKG) having a master public/secret key pair is responsible for generating private keys for users. This feature leads to an inherent key escrow problem: users' private keys are known to the PKG, therefore, it can decrypt any ciphertext and forge signatures on any message for any user. Al-Riyami and Paterson [1] introduced certificateless public key cryptography (CL-PKC) to solve the key-escrow problem. In CL-PKC, a user private key is a combination of some contribution of a KGC (called a partial private key) and some user-chosen secret, in such a way that the problem can be solved. CL-PKC is not purely ID-based, as a signature and a ciphertext are transmitted together with an additional user public key that is not required to be certified by any trusted authority. In order to verify a signed message, one must know both the user's identity and this additional public key.

Al-Riyami and Paterson [1] proposed a certificateless public-key encryption (CLE) scheme and a certificateless signature (CLS) scheme. Although the security model for CLE schemes was established in [1], while unforgeability of CLS schemes was not formally defined. Since Al-Riyami and Paterson's CLS scheme, several CLS schemes have been proposed [10,5,14]. They provided only informal analysis and were subsequently found to be vulnerable to key replacement attacks by type I adversaries [16,3,2]. Later, several formal security models have been proposed by presenting proven secure CLS schemes in the

E-mail address: kashim@nims.re.kr

http://dx.doi.org/10.1016/j.ins.2014.10.055
0020-0255/© 2014 Elsevier Inc. All rights reserved.

model assuming random oracles [9,17,4]. Liu et al. [11] proposed the first provably secure CLS scheme in the standard model, based on Water's scheme [13]. In addition to these direct constructions, there exist generic constructions that convert existing signature schemes in different infrastructures into CLS schemes. Yum and Lee [15] proposed a generic construction for CLS schemes by combining any standard signature (SS) scheme with any ID-based signature (IBS) scheme. Subsequently, Hu et al. [6] showed that this construction is insecure against key replacement attacks and then proposed its improved version by modifying the input of signing algorithm. In particular, Hu et al. [7] established a simplified definition and formal security models for CLS schemes which are shown to be more versatile than previous ones [9,17]. Au et al. [2] suggested a malicious-but-passive-KGC attack in which a KGC may not generate master public/secret key pair honestly to mount the attack, and they modified Hu et al.'s model for capturing the attack. They also showed that Al-Riyami and Paterson's scheme and its variants [1,9,10] are insecure against the malicious-but-passive-KGC attacks and the security of the CLS scheme converted from the modified Yum–Lee's construction is preserved in their new model. In summary, there are two types of adversaries In CLS schemes: a type I adversary represents a malicious third party who can replace a user public key, called a key replacement attack, and a type II adversary is a malicious KGC who knows the master secret, but cannot replace user public keys. Existence of type I adversary is due to the uncertified feature of a user public key and considering type II adversary is for solving the key escrow problem, i.e., disclosure of KGC's master secret does not compromise the secret of each user. In malicious-but-passive-KGC attacks, a type II adversary, KGC, is passive, in the sense that the KGC would not actively replace user public keys, and can generate a mater secret/public key pair to attack a target user. In this paper, we discuss differences between formal security models of CLS schemes in [17,2,7,8] and then point out security flaws on the restrictions of an adversary's final output in the security models by demonstrating key replacement attacks on three CLS schemes [10,4,11].

The rest of the paper is organized as follows. In Section 2, we describe a definition of CLS schemes and their formal security models proposed in [17]. We discuss differences in the security models in [17,2,8,7] in Section 3. In Section 4, we point out vulnerabilities of Liu et al.'s scheme, Choi et al.'s scheme and Li–Chen's scheme against key replacement attacks in Zhang et al.'s security model [17]. We then suggest improvements. Concluding remarks are given in Section 5.

2. Formal security models for CLS schemes

We briefly describe a definition for CLS schemes and their formal security models in [17].

COMPONENTS OF CERTIFICATELESS SIGNATURE SCHEMES. A CLS scheme $\mathcal{CLS} = (\textbf{Setup}, \textbf{Partial}-\textbf{Private}-\textbf{Key}-\textbf{Extract}, \textbf{Set}-\textbf{Secret}-\textbf{Value}, \textbf{Set}-\textbf{Private}-\textbf{Key}, \textbf{Set}-\textbf{Public}-\textbf{Key}, \textbf{CL}-\textbf{Sign}, \textbf{CL}-\textbf{Verify})$ is specified by six polynomial time algorithms with the following functionality:

- **Setup.** It takes as input a security parameter k, and returns a list `params` of system parameters and a master public/secret key pair (*mpk*,*msk*). The algorithm is assumed to be run by a KGC for the initial setup of a certificateless system.
- **Partial-Private-Key-Extract.** It takes as inputs `params`, a master secret key *msk* and a user identity $ID \in \{0,1\}^*$, and outputs a partial private key D_{ID}. This algorithm is run by the KGC once for each user, and the partial private key generated is assumed to be distributed securely to the corresponding user.
- **Set-Secret-Value.** Taking as inputs params and a user's identity *ID*, this algorithm generates a secret value S_{ID}. This algorithm is supposed to be run by each user in the system.
- **Set-Private-Key.** This algorithm takes params, a user's partial private key D_{ID} and his secret value S_{ID}, and outputs a full private key SK_{ID}. This algorithm is run by each user.
- **Set-Public-Key.** It takes as inputs `params` and a user's secret value S_{ID}, and generates a public key PK_{ID} for that user. This algorithm is run by the user, and the resulting public key is assumed to be publicly known.
- **CL-Sign.** This algorithm takes as inputs `params`, a message $m \in \{0,1\}^*$, a user's identity *ID*, and the user's full private key SK_{ID}, and outputs a signature σ.
- **CL-Verify.** This algorithm takes as inputs `params`, a public key PK_{ID}, a message m, a user's identity *ID*, and a signature σ, and returns a bit b. $b = 1$ means that the signature is accepted, whereas $b = 0$ means rejected.

There are two types of adversaries, $\mathcal{A}^I$ and $\mathcal{A}^{II}$ in CLS schemes. The type I adversary $\mathcal{A}^I$ is a malicious third party who compromises a user secret key or replaces a user public key, while $\mathcal{A}^I$ is given neither a master secret key *msk* nor a partial private key. The type II adversary $\mathcal{A}^{II}$ is a malicious KGC, who knows a master secret *msk* and hence can derive the value of any user's partial private key, while it can neither access to a user public key nor replace a user secret key.

UNFORGEABILITY OF CLS SCHEMES. Let $\mathcal{CLS}$ be a CLS scheme. We consider two games Game I and Game II where $\mathcal{A}^I$ and $\mathcal{A}^{II}$ interact with their challenger in these two games, respectively.

[Game I]. This is the game in which $\mathcal{A}^I$ interacts with the challenger.

- **Phase I-1:** The challenger runs $\texttt{Setup}(1^k)$ for generating (*mpk*,*msk*) and `params`. The challenger then gives `params` and *mpk* to $\mathcal{A}^I$ while keeping *msk*.
- **Phase I-2:** $\mathcal{A}^I$ performs the following oracle-query operations:
  - `Extract Partial Private Key Queries`: On receiving such a query, the challenger computes D_{ID}=`Partial-Private-Key-Extract`(`params`,*msk*, *ID*) and returns it to $\mathcal{A}^I$.

- `Extract Private Key Queries`: Upon receiving such a query, the challenger first computes D_{ID} = `Partial-Private-Key-Extract(params,`*msk*, *ID*) and then S_{ID}=`Set-Secret-Value(params,` *ID*) as well as SK_{ID} = `Set-Private-Key(Params,` D_{ID}, S_{ID}). It returns SK_{ID} to $\mathcal{A}^I$.
- `Request Public Key Queries`: Upon receiving such a query, the challenger computes D_{ID} = `Partial-Private-Key-Extract(params,`*msk*, *ID*), and S_{ID} = `Set-Secret-Value(params,`*ID*). It then computes PK_{ID} = `Set-Public-Key(params,` S_{ID}) and returns it to $\mathcal{A}^I$.
- `Replace Public Key Queries`: $\mathcal{A}^I$ may replace a public key PK_{ID} with a value chosen by him. It is not required for $\mathcal{A}^I$ to provide the corresponding secret value when making this query.
- `Signing Queries`: On receiving such a query, the challenger finds SK_{ID} from its query-answer list, computes σ = `CL-Sign(params,`M, ID, SK_{ID}), and returns it to $\mathcal{A}^I$. If the public key PK_{ID} has been replaced by $\mathcal{A}^I$, then the challenger cannot find SK_{ID} and thus the signing oracle's answer may be incorrect. In such case, we assume that $\mathcal{A}^I$ may additionally submit the secret information S_{ID} corresponding to the replaced public-key PK_{ID} to the signing oracle.

- **Phase I-3:** Finally, $\mathcal{A}^I$ outputs a forgery σ^* on M^* corresponding to ID^* and a public key PK_{ID^*} and wins the game if

- `CL-Verify(params,`$PK_{ID^*}, ID^*, M^*, \sigma^*) = 1$,
- ID^* cannot be an identity for which the private key has been extracted,
- ID^* cannot be an identity for which both the public key has been replaced and the partial private key has been extracted, and
- M^* should not be queried to the signing oracle with respect to ID^* and PK_{ID^*}.

[Game II]. This is a game in which $\mathcal{A}^{II}$ interacts with the challenger.

- **Phase II-1:** The challenger runs `Setup`(1^k) to generate (*mpk*,*msk*) and `params`. The challenger gives `params` and (*mpk*,*msk*) to $\mathcal{A}^{II}$.
- **Phase II-2:** $\mathcal{A}^{II}$ performs the following operations:
  - `Extract Private Key Queries`: On receiving such a query, the challenger computes D_{ID} = `Partial-Private-Key-Extract(params,`*msk*, *ID*), S_{ID} = `Set-Secret-Value(params,` *ID*) and SK_{ID} = `Set-Private-Key(Params,`D_{ID}, S_{ID}). It returns SK_{ID} to $\mathcal{A}^{II}$.
  - `Request Public Key Queries`: On receiving such a query, the challenger sets D_{ID}=`Partial-Private-Key-Extract(params,`*msk*,*ID*), S_{ID}=`Set-Secret-Value(params,`*ID*), and then computes PKID = Set-Public-Key Queries(params, SID, ID). It returns PK_{ID} to $\mathcal{A}^{II}$.
  - `Signing Queries`: On receiving such a query, the challenger finds SK_{ID} from its query-answer list, computes σ = `CL-Sign(params,`M, ID, SK_{ID}), and returns it to $\mathcal{A}^{II}$.
- **Phase II-3:** Finally, $\mathcal{A}^{II}$ outputs a forgery σ^* on M^* corresponding to ID^* and a public key PK_{ID^*} and wins the game if
  - `CL-Verify(params,`$PK_{ID^*}, ID^*, M^*, \sigma^*) = 1$,
  - ID^* has not been issued as a private key query and
  - M^* should not be queried to the signing oracle with respect to ID^* and PK_{ID^*} .

We define $\text{Succ}_{\mathcal{A}}(k)$ as the probability that an adversary $\mathcal{A}$ ($\mathcal{A}^I$ or $\mathcal{A}^{II}$) succeeds in the above games (Game I or Game II). If for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, the success probability $\text{Succ}_{\mathcal{A}}(k)$ is negligible, then we say that a CLS scheme is existentially unforgeable against chosen message attacks.

3. Differences in three formal security models

All the formal security models for CLS schemes can be merged to four ones in [17,2,7,8]. Here, we discuss differences in the formal security models.

(1) **Refined signing oracles in formal security models against type I and type II adversaries.** Huang et al. [8] divided the adversaries into three types: normal, strong and super.

- **Normal adversary.** It cannot obtain signatures related to a target identity and its replaced public keys.
- **Strong adversary.** It can obtain signatures related to a target identity and its replaced public keys by providing secret values corresponding to the replaced public keys for a challenger in the security model.
- **Super adversary.** It can obtain signatures related to a target identity and its replaced public keys without providing the secret values corresponding to the replaced public keys for the challenger. Security against the super adversary is the strongest among the notions.

Games between normal, strong, super-Type I and II games with a challenger are defined as in [7] by replacing the signing oracles with normal, strong, super-signing oracles, respectively.

(2) **Capturing malicious-but-passive KGC attacks in formal security model against a type II adversary.** Au et al. [2] proposed a new type II adversaries by capturing malicious-but-passive-KGC attacks in which a KGC may not generate master public/secret key pair honestly to mount the attack. Their model removes the assumption that KGC must be benign during the master key generation step and user partial private key generation. It allows KGC to choose a user to attack during the master key generation.

(3) **Formal security models against a type II adversary.** A main difference in the models in [17,7] is whether the models allow $\mathcal{A}^{II}$ to query ReplaceKey in Game II at any point or not. Of course, it is assumed that $\mathcal{A}^{II}$ does not mount an attack against a user with an identity *ID* when the user public key is also replaced by $\mathcal{A}^{II}$ because $\mathcal{A}^{II}$ can always get the user partial key. However, allowing $\mathcal{A}^{II}$ to query ReplaceKey with identities other than ID does not boost $\mathcal{A}^{II}$'s attacking capability either. This is because $\mathcal{A}^{II}$ can always obtain all the secrets corresponding to identities other than *ID*.

(4) **Formal security models against type I and type II adversaries.** Hu et al. [7] provided a simplified definition of CLS schemes and formal security models. First, they simplified the components of CLS scheme by requiring only five algorithms, (`MasterKeyGen`, `PartialKeyGen`, `UserKeyGen`, `CL-Sign`, `CL-Verify`) instead of seven algorithms in [1,17]. In the model [17], the restriction of $\mathcal{A}^{I}$'s final output in Game I is.

- ID^* has never queried Partial-Private-Key-Extract (RevealPartialKey),
- ID^* has never queried ReplacePublicKey nor Partial-Private-Key-Extract (RevealPartialKey), and
- m^* should not be queried to the signing oracle with respect to ID^* and PK_{ID^*}.

Hu et al. [7] simplified the restrictions above are simplified as.

- ID^* has never queried RevealPartialKey, and
- m^* should not be queried to the signing oracle with respect to ID^*.

A main difference between the model in [17] and other ones [7,8] is the restriction of an adversary's final output.

- Zhang et al.'s model [17] requires that the triple (m^*, ID^*, PK_{ID^*}) involved in the adversary's final output has never been queried to the signing oracle, while the models [6,2] replace the triple with the pair $(m^*,\ ID^*)$.

There is a great difference between these two restrictions in the following sense: Suppose that a type I adversary $\mathcal{A}^{I}$ has obtained a signature σ^* on m^* for $\{ID^*,\ PK_{ID^*}\}$ from the signing oracle. In this case, if $\mathcal{A}^{I}$ can obtain another signature σ' on the same message m^* for $\{ID^*,\ PK'_{ID^*}\}$, where PK'_{ID^*} is a new public key corresponding to ID^* replaced by $\mathcal{A}^{I}$, then a CLS scheme is insecure in Zhang et al.'s model, but it is still secure in the other models. The restriction of the adversary's final output that the pair $(m^*,\ ID^*)$ has never been queried to the signing oracle comes from the formal security model of ID-based signature schemes. However, it is not suitable for security of CLS schemes, as a user public key corresponding to an identity can be replaced with a new public key chosen by the adversary at any time. Thus, the signatures for (m, ID, PK_{ID}) and (m, ID, PK'_{ID}) should be differentiated. We will present key replacement attacks on three CLS schemes [10,4,11] associated with these restrictions in next section.

4. Security pitfalls on the restriction of adversary's final output

4.1. Liu et al.'s CLS scheme

Liu et al. [11] proposed the first provably secure CLS scheme in the standard model by using Waters' technique [13]. They follow the definition of CLS schemes in [6] and the formal security models in [17].

4.1.1. Liu et al.'s CLS scheme

- **Setup.** Given a security parameter $k \in \mathbb{Z}^+$, the algorithm works as follows;
  – Choose two collision-resistant hash functions $H_u : \{0,1\}^* \rightarrow \{0,1\}^{n_u}$ and $H_m : \{0,1\}^* \rightarrow \{0,1\}^{n_m}$ for some $n_u,\ n_m \in \mathbb{Z}$. They are used to create identities and messages of the desired length.
  – Select an admissible pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, pick random values $\alpha \in_R \mathbb{Z}_p^*, g_2 \in_R \mathbb{G}_1$ and compute $g_1 = g^\alpha$, where g is a generator of $\mathbb{G}_1$.
  – Choose randomly the following elements:
  $$u', m' \in_R \mathbb{G}_1,\ \hat{u}_i \in_R \mathbb{G}_1,\ i = 1, \ldots, n_u,\ \hat{m} \in_R \mathbb{G}_1,\ i = 1, \ldots, n_m.$$
  Let $\widehat{U} = \{\hat{u}_i\}$ and $\widehat{M} = \{\hat{m}_i\}$. The public parameter is $\texttt{Params} = \langle p,\ \mathbb{G}_1,\ \mathbb{G}_2,\ e,\ g,\ g_1,\ g_2,\ u',\ \widehat{U},\ m',\ \widehat{M}\rangle$ and the master secret is g_2^α.
- **Partial-Private-Key-Extract.** Given an identity *ID*, compute $\mathtt{u} = H_u(ID)$. Let $u[i]$ be the i-th bit of $\mathtt{u}$. Define $\mathcal{U} \subset \{1, \ldots, n_u\}$ to be the set of indices such that $u[i] = 1$. Choose a random number $r_u \in \mathbb{Z}_p^*$ and compute $U = u'\prod_{i\in\mathcal{U}}\hat{u}_i$ and a partial private key pair $(\mathsf{psk}^{(1)}, \mathsf{psk}^{(2)}) = (g_2^\alpha \cdot U^{r_u},\ g^{r_u})$.
- **User-Key-Generation.** Given an identity *ID*, pick a random $x \in \mathbb{Z}_p^*$ and return a public key pair $(\mathsf{pk}^{(1)}, \mathsf{pk}^{(2)}) = (g^x,\ g_1^x)$ and a secret key $\mathsf{sk} = x$.
- **CL-Sign.** Given a partial private key pair $(\mathsf{psk}^{(1)}, \mathsf{psk}^{(2)})$, a secret key *sk*, an identity *ID*, a public key pair $(\mathsf{pk}^{(1)}, \mathsf{pk}^{(2)})$ and a message m, compute $\mathtt{m} = H_m(m)$. Let $m[i]$ be the i-th bit of $\mathtt{m}$ and $\mathcal{M} \subset \{1, \ldots, n_m\}$ be the set of indices i such that $\mathtt{m}[i] = 1$. Choose random numbers $r_\pi,\ r_m \in \mathbb{Z}_p^*$ and compute $U = u'\prod_{i\in\mathcal{U}}\hat{u}_i$ and
$$\sigma = \left(\left(\mathsf{psk}^{(1)}\right)^{\mathsf{sk}} U^{r_\pi} \left(m' \prod_{i\in\mathcal{M}} \hat{m}_i \right)^{r_m},\ \left(\mathsf{psk}^{(2)}\right)^{\mathsf{sk}} g^{r_\pi}, g^{r_m} \right)$$

- **CL-Verify.** Given a signature $\sigma = (V, R_\pi, R_m)$ for $\{ID,\ (\mathsf{pk}^{(1)},\ \mathsf{pk}^{(2)})\}$ on a message m, compute $\mathtt{m} = H_m(m),\ U = u'\prod_{i\in\mathcal{U}}\hat{u}_i$ and check whether the equations

$$e(\mathsf{pk}^{(1)}, g_1) = e(\mathsf{pk}^{(2)}, g)\ \ e(V, g) = e(g_2, \mathsf{pk}^{(2)} e(U, R_\pi) e\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i, R_m\right),$$

hold or not. Output valid if both equalities hold. Otherwise output invalid.

4.1.2. Key Replacement attack on liu et al.'s CLS scheme

Suppose that a type I adversary $\mathcal{A}^I$ wants to forge a signature of a user A in the scheme. via the signing oracle, $\mathcal{A}^I$ has obtained a signature $\sigma = (V, R_\pi, R_m)$ on m for $\{ID_A,\ (\mathsf{pk}^{(1)}, \mathsf{pk}^{(2)})\}$, where

$$\sigma = \left(\left(\mathsf{psk}^{(1)}\right)^x U^{r_\pi}\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i\right)^{r_m},\ \left(\mathsf{psk}^{(2)}\right)^x g^{r_\pi}, g^{r_m}\right)$$

and $(\mathsf{pk}^{(1)}, \mathsf{pk}^{(2)}) = (g^x, g_1^x)$. Then $\mathcal{A}^I$ can forge $\sigma' = (V', R'_\pi, R'_m)$ on the same message m for another public key pair $(\overline{\mathsf{pk}}^{(1)},\ \overline{\mathsf{pk}}^{(2)})$ corresponding to ID_A of its choice as follows:

- First, $\mathcal{A}^I$ chooses a random $y \in_R \mathbb{Z}_p^*$ and computes a new public key pair $(\overline{\mathsf{pk}}^{(1)}, \overline{\mathsf{pk}}^{(2)}) = ([\mathsf{pk}(1)]^y, [\mathsf{pk}^{(2)}]^y) = (g^{xy}, g_1^{xy})$ being replaced.
- Next, $\mathcal{A}^I$ computes $\sigma' = (V', R'_\pi, R'_m)$ as follows:

$$V' = V^y = \left(\mathsf{psk}^{(1)}\right)^{xy} U^{r_\pi y}\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i\right)^{r_m y}$$

$$R'_\pi = R_\pi^y = \left(\mathsf{psk}^{(2)}\right)^y g^{r_\pi y} = g^{r_u xy + r_\pi y},\ R'_m = (R_m)^y = g^{r_m y}.$$

Then $\sigma' = (v', r'_\pi, R'_m)$ is a valid signature on m for $\{ID, (\overline{\mathsf{pk}}^{(1)}, \overline{\mathsf{pk}}^{(2)})\}$ since it satisfies the verification equations:

$$\begin{aligned} e(V, g) &= e\left(\left[g_2^\alpha\left(u'\prod_{i\in\mathcal{U}}\hat{u}_i\right)^{r_u}\right]^{xy}\left(u'\prod_{i\in\mathcal{U}}\hat{u}_i\right)^{r_\pi y}\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i\right)^{r_m y},\ g\right) \\ &= e(g_2^{\alpha xy}, g)\cdot e\left(\left(u'\prod_{i\in\mathcal{U}}\hat{u}_i\right)^{r_u xy + r_\pi y}, g\right)\cdot e\left(\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i\right)^{r_m y}, g\right) \\ &= e(g_2, g_1^{xy})\cdot e(U, g^{r_u xy + r_\pi y})\cdot e\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i, g^{r_m y}\right) \\ &= e(g_2, \overline{\mathsf{pk}}^{(2)})\cdot e(U, R'_\pi)\cdot e\left(m'\prod_{i\in\mathcal{M}}\hat{m}_i, R'_m\right). \end{aligned}$$

Consequently, the adversary succeeds in forging A's signature for the replaced public key $(\overline{\mathsf{pk}}^{(1)},\ \overline{\mathsf{pk}}^{(2)})$ in Zhang et al.'s model, as $(ID^*, m^*, \overline{PK_{ID^*}})$ has never been queried to the signing oracle.

4.2. Choi et al.'s CLS scheme

4.2.1. Choi et al.'s CLS scheme

- **Setup.** Given a security parameter $k \in \mathbb{Z}$, the algorithm works as follows;
  - Run the parameter generator $\mathcal{G}$ on input k to generate a prime q, two groups $\mathbb{G}_1$, $\mathbb{G}_2$ of order q, a generator P in $\mathbb{G}_1$ and an admissible pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
  - Pick a random $s \in \mathbb{Z}_q^*$ and set $P_{Pub} = sP$, where s is a master secret.
  - Choose cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1, H_2 : \mathbb{G}_1 \to \mathbb{Z}_q^*$ and $H_3 : \{0,1\}^* \to \mathbb{Z}_q^*$. The system parameter is $\mathtt{Params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, P, P_{Pub}, H_1, H_2, H_3\rangle$.
- **Partial-Private-Key-Extract.** On input the system parameter Params, a master secret s, and identity ID_A, output a partial private key $D_A = s\cdot Q_A$ corresponding to ID_A, where $Q_A = H_1(ID_A)$.
- **Set-Secret-Value.** On input a security parameter k and an identity ID_A, choose a random value $x_A \in \mathbb{Z}_q^*$ and return x_A as ID_A's secret value.
- **Set-Public-Value.** On input a secret key x_A, compute $PK_A = x_AP$ and return the public key PK_A.
- **Set-Private-Key.** On input x_A, PK_A, and D_A, compute $y_A = H_2(PK_A)$ and $SK_A = \frac{1}{x_A + y_A}\cdot D_A$ and return the (full) private key SK_A.
- **CL-Sign.** On input ID_A, SK_A, and a message m, perform the following steps:
  - Choose $r \in_R \mathbb{Z}_q$ and compute $U = r\cdot Q_A$, where $Q_A = H_1(ID_A)$.
  - Compute $h = H_3(m, U)$ and $V = (r + h)\cdot SK_A$.
  - Set $\sigma = (U, V)$ as a signature on m for $\{ID_A, PK_A\}$.

- **CL-Verify.** On input `params`, ID_A, and a signature $\sigma = (U, V)$ on m for $\{ID_A, PK_A\}$,
  - Compute $Q_A = H_1(ID_A), y_A = H_2(PK_A)$, and $h = H_3(m, U)$.
  - Check if $e(V, PK_A + y_A P) = e(U + hQ_A, P_{Pub})$ holds. If the equation holds, it outputs 1, otherwise 0.

Now, we show that Choi et al.'s scheme is insecure against a type I adversary in Zhang et al.'s model [17].

4.2.2. Key replacement attack on Choi et al.'s CLS scheme

Suppose that a type I adversary $\mathcal{A}^I$ wants to forge a signature of a user A with an identity ID_A in the scheme.

- First, $\mathcal{A}^I$ chooses x_A and computes $PK_A = x_A P$. via the `CL-Sign` oracle, $\mathcal{A}^I$ can obtain a A's signature $\sigma = (U, V)$ on m for $\{ID_A, PK_A\}$, where $U = r \cdot Q_A$ and $V = (r + h) \cdot SK_A$. Then $\mathcal{A}^I$, who knows x_A, computes

$$(x_A + y_A) \cdot V = (x_A + y_A)(r + h) \cdot SK_A = (x_A + y_A)(r + h)\frac{1}{x_A + y_A}D_A = (r + h)D_A.$$

- Finally, $\mathcal{A}^I$ can forge $\sigma' = (U', V')$ on the same message m for another replaced public key $PK'_A = x'_A P$, where

$$U' = U, \quad V' = \frac{1}{x'_A + y'_A}(r + h)D_A = (r + h)\frac{1}{x'_A + y'_A}D_A,$$

as $\frac{1}{x'_A + y'_A}D_A$ is the private key SK_A of ID_A corresponding to the replaced public key $PK'_A = x_A P$, where $y'_A = H_2(PK'_A)$. In this case, the triple $(m,\ ID_A,\ PK'_A)$ has never requested to the signing oracle, while $(m,\ ID_A)$ has requested to the oracle. The adversary succeeds in forging A's signature for the replaced public key PK'_A. Therefore, it is insecure against the key replacement attack in Zhang et al.'s model, but it remains to be secure in other models.

4.3. Li–Chen's CLS scheme

Li–Chen's CLS scheme [10] also follows Al-Riyami–Paterson's definition, security models and the private key generation method [1]. The scheme runs as follows.

4.3.1. Li–Chen's CLS scheme

- **Setup.** Given a security parameter $k \in \mathbb{Z}$, the algorithm works as follows;
  - Run the parameter generator $\mathcal{G}$ on input k to generate a prime q, two groups $\mathbb{G}_1$, $\mathbb{G}_2$ of order q, a generator P in $\mathbb{G}_1$ and an admissible pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
  - Pick a random $s \in \mathbb{Z}_q^*$ and set $P_{Pub} = sP$.
  - Choose cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1, H_2 : \mathbb{G}_1 \to \mathbb{Z}_q$. The system parameter is `Params` = $\langle q, \mathbb{G}_1, \mathbb{G}_2, e, P, P_{Pub}, H_1, H_2 \rangle$.
- **Partial-Private-Key-Extract.** For an identity ID_A and a master key s, compute $Q_A = H_1(ID_A)$ and return $D_A = s \cdot Q_A$ as a partial private key for ID_A.
- **Set-Secret-Value.** For an identity ID_A, choose $x_A \in \mathbb{Z}_q$ and output x_A as a secret value of ID_A.
- **Set-Private-Key.** To construct a full private key for Alice, compute $S_A = x_A \cdot D_A = x_A \cdot (sQ_A)$ as its full private key.
- **Set-Public-Value.** For an identity ID_A, compute $X_A = x_A \cdot P, Y_A = x_A \cdot P_{Pub}$, and set $P_A =< X_A, Y_A >$ as its public key.
- **CL-Sign.** Given a message m and a private key S_A,
  - Choose $r \in_R \mathbb{Z}_q$ and compute $U = r \cdot Q_A$, where $Q_A = H_1(ID_A)$.
  - Compute $h = H_2(m, U)$ and $V = (r + h) \cdot S_A$.
  - Set $\sigma = (U, V)$ as a signature on m for $\{ID_A, (X_A, Y_A)\}$.
- **CL-Verify.** To verify a signature $\sigma = (U, V)$ on m for $\{ID_A, (X_A, Y_A)\}$, follows;
  - Check whether $e(X_A, P_{Pub}) = e(Y_A, P)$ holds or not. If not, stop and reject the signature; otherwise, continue.
  - Compute $h = H_2(m, U)$ and check whether $e(P, V) = e(Y_A, h \cdot Q_A + U)$ holds or not. If not, stop and reject the signature; otherwise, accept it.

4.3.2. Key replacement attack on Li–Chen's CLS scheme

Suppose that a type I adversary $\mathcal{A}^I$ wants to forge a signature of a user A with an identity ID_A in the scheme. We assume that $\mathcal{A}^I$ has already obtained a A's signature $\sigma = (U, V)$ on m for $\{ID_A, (X_A, Y_A)\}$, where $U = r \cdot Q_A$ and $V = (r + h) \cdot S_A$. Then $\mathcal{A}$ can forge $\sigma' = (U', V')$ on the same message m for another public key pair (X'_A, Y'_A) corresponding to ID_A as follows;

- ID_A chooses a random $t \in \mathbb{Z}_q^*$, computes a new public key pair $(X'_A = t \cdot X_A, Y'_A = t \cdot Y_A)$ being replaced.
- Next, $\mathcal{A}^I$ computes $U' = U = r \cdot Q_A$ and $V' = t \cdot V$. In this case, the hash value $h = H_2(m, U)$ is invariant because the message m and the random part U of the signature are not changed. Then $\sigma' = (U', V')$ is a valid signature on m for $\{ID_A, (X'_A, Y'_A)\}$ since

$$V' = t \cdot V = t \cdot (r + h) \cdot S_A = (r + h) \cdot t \cdot S_A = (r + h) \cdot S'_A,$$

where $S'_A = t \cdot S_A = tx_A \cdot D_A$ is a valid full private key of A with respect to the replaced public key pair (X'_A, Y'_A). Therefore, the adversary succeeds in forging A's signature for the replaced public key. Therefore, it is insecure against the key replacement attack in Zhang et al.'s model, but it is still secure in other models.

4.4. Discussion and improvements

In the certificateless setting, an adversary can replace user's public keys with new public keys of its own choice. An adversary, who has obtained a user's signature on a message for a user public key PK_{ID} (in fact, the adversary can access to the signing oracle to obtain the signature), cannot forge a new signature on the same message for another user public key PK'_{ID} corresponding to ID. This is related to the restriction of the adversary's final output in the security models. The attacks show that the three CLS schemes [10,4,11] are insecure in a model, but they remain secure on other models. Thus, to reflect adversaries' power to replace users' public keys in a realistic way, the restriction of the adversary's final output in the models [2,7,8] should be replaced that the triple (ID^*, m^*, PK_{ID^*}) has never requested to the signing oracle.

Improvements are concerned with explicitness of signature itself. Signatures should be possible for a recipient to confirm that they are correctly bound to a signer's identity, public key, message and random values in signature verification. This is normally achieved by providing redundancy in the input of a hash function if hash function is used. Adding redundant information in signed messages can prevent many attacks caused by the lack of explicitness in cryptographic messages. Thus, cryptographic messages should contain sufficient redundancy, while superfluous redundancy should be removed. In Choi et al.'s scheme and Li–Chen's scheme, we replace $H_3(m, U)$ with $H_3(m, U, ID, PK_{ID})$ in **CL-Sign** by adding $\{ID, PK_{ID}$ to the input of a hash function if hash function, where the signatures are of the form $\sigma = (U, V)$ such that $U = rQ_A$ and $V = (r + h)SK_A$. In Liu et al.'s scheme, $H_m(m, U, ID, PK_{ID}) = m' \prod_{i \in \mathcal{M}} \hat{m}_i$. This makes it possible to bind σ to the corresponding identity, user public key, random value, and message. Thus, it can prevent the key replacement attacks. Security proofs of the original CLS schemes against type I and II adversaries in the model [7,8] can be preserved for these improved CLS schemes in Zhang et al.'s model [17] by simply modifying H_3-hash queries and CL-Signing queries.

5. Conclusion

We have discussed the differences in the formal security models in [17,2,7,8]. We have pointed out security pitfalls on the restrictions of an adversary's final output in the security models by demonstrating key replacement attacks on the three CLS schemes [10,4,11]. We have showed that Liu et al.'s, Choi et al.'s and Li–Chen's CLS schemes are insecure against key replacement attacks in Zhang et al.'s model [17], but they remain secure in the other models. Thus, to reflect adversaries' power to replace users' public keys in a realistic way, the restriction of the adversary's final output in the models [2,7,8] should be replaced that the triple (ID^*, m^*, PK_{ID^*}) has never requested to the signing oracle.

References

[1] S.S. Al-Riyami, K.G. Paterson, Certificateless public key cryptography, in: Advances in Cryptography: Asiacrypt'03, LNCS, vol. 2894, Springer-Verlag, 2003, pp. 452–473.

[2] M.H. Au, J. Chen, J.K. Liu, Yi Mu, Duncan S. Wong, G. Yang, Malicious KGC attacks in certificateless cryptography, in: ASIACCS'07, 2007, pp. 302–311.

[3] X. Cao, K.G. Paterson, W. Kou, An Attack on a Certificateless Signature Scheme, Cryptology ePrint Archive: Report 2006/367.

[4] K.Y. Choi, J.H. Park, J.Y. Hwang, D.H. Lee, Efficient certificateless signature schemes, in: ACNS'07, LNCS, vol 4521, 2007, pp. 443–458.

[5] M.C. Gorantla, A. Saxena, An efficient certificateless signature scheme, in: CIS'05, LNAI, vol. 3802, Springer-Verlag, 2005, pp. 110–116.

[6] B.C. Hu, D.S. Wong, Z. Zhang, X. Deng, Key replacement attack against a generic construction of certificateless signature, in: ACISP'06, LNCS, vol. 4058, Springer-Verlag, 2006, pp. 235–246.

[7] B.C. Hu, D.S. Wong, Z. Zhang, X. Deng, Certificateless signature: a new security model and an improved generic construction, Design. Code. Cryptogr. 42 (2) (2007) 109–126.

[8] X. Huang, Y. Mu, W. Susilo, D.S. Wong, W. Wu, Certificateless signatures: new schemes and security models, Comput. J. 55 (4) (2012) 457–474.

[9] X. Huang, W. Susilo, Y. Mu, F. Zhang, On the security of certificateless signature schemes from Asiacrypt 2003, in: CANS'05, LNCS, vol. 3810, Springer-Verlag, 2005, pp. 13–25.

[10] X. Li, K. Chen, L. Sun, Certificateless signature and proxy signature schemes from bilinear pairings, Lithuanian Math. J. 45 (1) (2005) 76–83.

[11] J.K. Liu, M.H. Au, W. Susilo, Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model, in: ASIACCS'07, 2007, pp. 273–283.

[12] A. Shamir, Identity-based cryptosystems and signature schemes, in: Advances in Cryptography: Crypto'84, LNCS, vol. 196, Springer-Verlag, 1984, pp. 47–53.

[13] B. Waters, Efficient identity-based encryption without random oracles, in: Advances in Cryptology-Eurocrypt'05, LNCS, vol. 3494, Springer-Verlag, 2005, pp. 114–127.

[14] W.S. Yap, S.H. Heng, B.M. Goi, An efficient certificateless signature scheme, in: Emerging Directions in Embedded and Ubiquitous Computing, EUCWorkshops 2006, LNCS, vol. 4097, Springer-Verlag, 2006, pp. 322–331.

[15] D.H. Yum, P.J. Lee, Generic construction of certificateless signature, in: ACISP'04, LNCS, vol. 3108, Springer-Verlag, 2004, pp. 200–211.

[16] Z. Zhang, D. Feng, Key Replacement Attack on a Certificateless Signature Scheme, Cryptology ePrint Archive: Report 2006/453.

[17] Z. Zhang, D. Wong, J. Xu, D. Feng, Certificateless public-key signature: security model and efficient construction, in: ACNS'06, LNCS, vol. 3989, Springer-Verlag, 2006, pp. 293–308.