Certificateless Public Key Encryption Secure against Malicious KGC Attacks in the Standard Model

Yong Ho Hwang
(Software Laboratories, Samsung Electronics Co., LTD, Korea
yongh.hwang@samsung.com)

Joseph K. Liu
(Institute for Infocomm Research (I^2R), Singapore
ksliu@i2r.a-star.edu.sg)

Sherman S.M. Chow
(New York University, USA
schow@cs.nyu.edu)

Abstract: Recently, Au *et al.* [Au et al. 2007] pointed out a seemingly neglected security concern for certificateless public key encryption (CL-PKE) scheme, where a malicious key generation center (KGC) can compromise the confidentiality of the messages by embedding extra trapdoors in the system parameter. Although some schemes are secure against such an attack, they require random oracles to prove the security. In this paper, we first show that two existing CL-PKE schemes without random oracles are not secure against malicious KGC, we then propose the first CL-PKE scheme secure against malicious KGC attack, with proof in the standard model.

Key Words: certificateless encryption, malicious KGC attack, standard model

Category: E.3

1 Introduction

In the traditional Public Key Cryptosystem (PKC), a trusted party called the Certification Authority (CA) issues a digitally signed certificate binding the identity and the public key of a user. However, the need for public key infrastructure supporting certificates is considered the main difficulty on the deployment and management of a traditional PKC. Identity Based Cryptography (IBC), envisioned in [Shamir 1984], solved this problem by using any string (such as email address, user name or phone number) as public keys, where a trusted third party called the Private Key Generator (PKG) manages the generation and distribution of all system parameters including the user's private key. While the complexity involved with the certificates is gone, it has the key escrow problem since the PKG generates the private key of every user.

Certificateless Public Key Cryptography (CL-PKC) is an intermediate between IBC and PKC [Al-Riyami and Paterson 2003]. Its main purpose is to solve the key escrow problem inherited from IBC without the use of certificates as in the traditional PKC. In CL-PKC, a Key Generation Center (KGC) is involved in issuing *user partial private key* computed from the master secret. The user also *independently* generates an additional user private key and the corresponding user public key. Cryptographic operations such as decryption and signature generation can then be performed successfully only when both the user partial private key and the user private key are known. Knowing only one of them should not be able to impersonate the user, that is, carrying out any cryptographic operations for a user. In this way, even if the KGC knows the user partial private key, impersonation is not possible. Since the KGC is no longer fully trusted, we thus have two different types of CL-PKE adversaries:

Type I. The adversary gets the user private key and/or *replaces* the user public key with some value chosen by the adversary. However, it does not know the user partial private key and the master secret.

Type II. The adversary knows the master secret, but does not know the user private key or being able to replace the user public key.

1.1 Certificateless Encryption Schemes in the Standard Model

Since CL-PKE offers both the advantages of identity-based encryption (IBE) for having no certificate, and public key encryption (PKE) for being free from key escrow, it is natural to build a CL-PKE scheme using an IBE scheme and a PKE scheme. Some generic constructions [Yum and Lee 2004a, Yum and Lee 2004b] take this approach, but are later shown to be insecure [Galindo et al. 2006]. Some generic constructions [Bentahar et al. 2005, Libert and Quisquater 2006] actually require on the random oracles for the formal security analysis. A generic construction without random oracles, when instantiated by IBE and PKE in the standard model, gives a CL-PKE in the standard model.

Chow, Boyd and Gonzalez Nieto [Chow et al. 2006] introduced the paradigm of Security-Mediated Certificateless Encryption (SMCE), which is a generalization of CL-PKE. In SMCE, the user partial private key is not issued directly to the user, but to a security-mediator (SEM). For decryption, the user needs to ask the SEM to perform the partial decryption first, and gets the message back by a further decryption using the user private key. In this way, instant revocation is possible by instructing the SEM to stop answering any further decryption requests for revoked users. When two partial decryption algorithms in SMCE are combined into one, it gives back the normal CL-PKE scheme.

In [Chow et al. 2006], a concrete construction in the random oracle model, and a generic construction from IBE and PKE without random oracles, are proposed. Only the concrete construction is secure against the Type I adversary. The generic construction is secure in a weaker sense (Type I^- adversary, see Section 2).

A concrete construction secure only under the same attack model as the generic construction in [Chow et al. 2006] is proposed in [Liu et al. 2007]. While this construction does not follow the generic approach in [Chow et al. 2006], it seems that there is no efficiency gain when compared with the generic construction in [Chow et al. 2006] instantiated with PKE and IBE in the standard model Recently, [Dent et al. 2008] proposed the first CL-PKE secure against the usual Type I adversary in the standard model.

1.2 Malicious KGC Attack

Recently, [Au et al. 2007] pointed out a seemingly neglected security concern in the Type II security model [Al-Riyami and Paterson 2003, Bentahar et al. 2005, Dent 2006, Galindo et al. 2006, Libert and Quisquater 2006], which implicitly assumes the KGC always generates the system parameters *honestly* according to the scheme specification. The model only allows the Type II adversary to know the master secret key of the KGC but not to choose the system parameter. The resulting scheme in such a model is definitely *not* strong enough to defend against a malicious KGC which may try every effort to break the system. In particular, a Type II adversary should be allowed to generate the key pair in any way it favours, which matches better with the original spirit of the CL-PKC.

Unfortunately, most of the existing schemes only focus on the study of Type I security, but neglect the significance of the malicious KGC (Type II security). A generic CL-PKE scheme [Libert and Quisquater 2006] is shown to be secure against malicious KGC attacks in [Au et al. 2007]. However, it requires the random oracle in the security proof.

Even all these generic constructions [Bentahar et al. 2005, Chow et al. 2006, Libert and Quisquater 2006] do not aim for achieving security against malicious KGC attack, one may think it is easy for a generic construction to achieve such a level of security as long as the PKE and the IBE are used in blackbox manners, and hence the decryption of the PKE part should be independent of that of the IBE part. However, a similar argument cannot be applied in concrete constructions. Indeed, we will show later that [Liu et al. 2007] and [Dent et al. 2008] are insecure in this sense. In addition, we propose the first concrete CL-PKE scheme secure against malicious KGC attack with proof in the standard model.

Organization. The rest of the paper is organized as follow. In Section 2, notations, preliminaries and the security model are reviewed. Section 3 shows that two existing CL-PKE schemes without random oracles are not secure against malicious KGC. We then propose the first CL-PKE scheme secure against malicious KGC attack. with proof in the standard model, in Section 4.

2 Preliminaries

2.1 Notation

If $k \in \mathbb{N}$, 1^k denotes the string of k ones. Let $\mathcal{A}$ be a probabilistic polynomial-time (PPT) algorithm. We use $s \leftarrow \mathcal{A}(x)$ denotes $\mathcal{A}$ outputs a string s on input of x. Note that x may be a vector having more than one element. We write $\mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots}(x)$ to indicate that $\mathcal{A}$ is an algorithm with input x and access to $\mathcal{O}_1, \mathcal{O}_2, \ldots$ oracles. Similarly, $z \leftarrow \mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots}(x)$ the operation of running $\mathcal{A}$ with inputs x and access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ and letting z be the output.

For a finite set X, $x \leftarrow X$ denotes the algorithm that samples an element uniformly at random from X. If w is neither an algorithm nor a set then $x \leftarrow w$ is a simple assignment statement. For a probability space P, $x \leftarrow P$ denotes the algorithm that samples a random element according to P. If $p(\cdot,\cdot,\cdots)$ is a boolean function, then $\Pr[p(x_1, x_2, \ldots)|x_1 \leftarrow P_1, x_2 \leftarrow P_2, \ldots]$ denotes the probability that $p(x_1, x_2, \ldots)$ is true after executing $x_1 \leftarrow P_1, x_2 \leftarrow P_2, \ldots$. Finally, a function $\epsilon : N \to R$ is negligible if for every constant $c > 0$ there exists an integer n_c such that $\epsilon(n) < n^{-c}$ for all $n \geq n_c$.

2.2 Certificateless Public Key Encryption

We review the definition and security notions of CL-PKE. We use the simplified model as used in [Au et al. 2007, Hu et al. 2006, Liu et al. 2007] which is slightly different from the original one [Al-Riyami and Paterson 2003]. The discussion of the main difference can be referred to [Hu et al. 2006].

Definition 1. A *certificateless public key encryption* is specified by 5 algorithms:

- Setup is a *setup algorithm* run by a key generation center (KGC). It takes a security parameter 1^k as input and returns the system parameters mpk and the master key msk. Intuitively, the system parameters will be publicly known, while the master secret key will be known only to the KGC.
- PSK is a *partial private key extraction algorithm* run by the KGC. It takes as inputs mpk, msk, and an identity ID of a user. It returns a partial private key psk_{ID} to a user with an identity ID.
- $\mathsf{UKeyGen}$ is a *user key generation algorithm* that takes as inputs mpk and psk_{ID}, and outputs the public/private key pair $(pk_{\mathsf{ID}}, sk_{\mathsf{ID}})$.
- Enc is an *encryption algorithm* that takes as inputs mpk, ID, pk_{ID}, and a message $m \in \mathcal{M}$ where $\mathcal{M}$ is a message space. It returns a ciphertext C.
- Dec is a *decryption algorithm* that takes as inputs mpk, sk_{ID}, and a ciphertext C. It returns a message $m \in \mathcal{M}$ if the ciphertext is valid and $\perp$ otherwise.

Correctness property requires that $\mathsf{Dec}(mpk, sk_{\mathsf{ID}}, \mathsf{Enc}(mpk, \mathsf{ID}, pk_{\mathsf{ID}}, m)) = m$.

Security Model. An adversary $\mathcal{A}$ is allowed to access to the following oracles.

– PKO. A *public key broadcast oracle* takes as input an identity ID and returns $\mathsf{pk_{ID}}$.
– RepO. A *public key replacement oracle* takes as input an identity ID and a valid public key $\mathsf{pk_{ID}}$. It replaces the associated user's public key with the new one $\mathsf{pk'_{ID}}$.
– PSKO. A *partial private key extraction oracle* takes as input an identity ID and returns $\mathsf{psk_{ID}}$.
– SKO. A *private key extraction oracle* takes as input an identity ID and returns $\mathsf{sk_{ID}}$ if the associated public key with ID was not replaced.
– DecO. A *decryption oracle* takes as inputs an identity ID and a ciphertext C, returns the decrypted ciphertext using $\mathsf{sk_{ID}}$. If the user's public key has been replaced, it requires an additional input of the corresponding private key. If it is not given or the ciphertext is invalid, $\perp$ will be returned.

First we consider the most common security goal and attack model: *Indistinguishability against adaptive chosen ciphertext attacks* (IND-CL-CCA). For an efficient algorithm $\mathcal{A}$, which runs in two stages $(\mathcal{A}_1, \mathcal{A}_2)$, we define the adversary's advantage as

$$\mathsf{Adv}_{\mathcal{A}_X^{\mathsf{CL}}}(k) = \left| \Pr\left[\mathfrak{b} = \mathfrak{b}' \middle| \begin{array}{l} (mpk, msk) \leftarrow \mathsf{Setup}(1^k), \\ (m_0, m_1, \mathsf{ID}^*, s) \leftarrow \mathcal{A}_1^{\mathcal{O}_X}(mpk, a), \\ \mathfrak{b} \leftarrow \{0,1\}, C^* \leftarrow \mathsf{Enc}(m_{\mathfrak{b}}, pk_{\mathsf{ID}^*}, \mathsf{ID}^*, mpk), \\ \mathfrak{b}' \leftarrow \mathcal{A}_2^{\mathcal{O}_X}(C^*, s) \end{array} \right] - \frac{1}{2} \right|$$

where $\mathcal{O}_{\mathrm{I}} = \{\mathsf{PSKO}, \mathsf{PKO}, \mathsf{RepO}, \mathsf{SKO}, \mathsf{DecO}\}$, $\mathcal{O}_{\mathrm{II}} = \{\mathsf{PKO}, \mathsf{SKO}, \mathsf{DecO}\}$ (i.e. X is I or II), and s is some internal state information. If $\mathcal{A}$ is a Type I adversary, $a = \emptyset$. Otherwise, $a = msk$. Actually, $\mathcal{A}_{II}$ does not need to issue partial private key queries, since it can compute them from the master key by itself. There are some restrictions as follows.
– $\mathcal{A}_I, \mathcal{A}_{II}$ cannot extract the private key for ID^*.
– $\mathcal{A}_I, \mathcal{A}_{II}$ cannot make a decryption query on C^*.
– $\mathcal{A}_I$ cannot request the private key for any identity whose public key has already been replaced.
– $\mathcal{A}_I$ cannot extract the partial private key for ID^* if it has replaced the public key for ID^* before the challenge phase.

In the Type I security model, we do not allow the Type I adversary to issue decryption queries made on identities for which it has replaced the public keys and the corresponding private key is not given. However, the strongest security model of [Al-Riyami and Paterson 2003] does expect that the decryption oracle should be able to output consistent answers even for identities whose public keys have been replaced and for which they do not know the corresponding private keys. This is a very strong notion of security. Generic schemes like [Bentahar et al. 2005, Chow et al. 2006] usually cannot afford such a strong attack, and have weakened this definition that the adversary does not make the decryption queries for which the public key has been replaced. We adopt this model for the Type I security. It is also referred as Type I$^-$ security in [Bentahar et al. 2005]. The rest of the paper omits the $^-$ sign.

In previous work, to simulate an honest-but-curious KGC, $\mathcal{A}_{II}$ is given msk. In the above model, we assume that the KGC honestly generates all the public parameters. However, we need a stronger security model to capture the actions of a malicious KGC, since it can maliciously compute the public parameters. In a real environment, the KGC generates the public parameters and the master secret key by itself, while in a security game the simulation algorithm $\mathcal{B}$ generates the public parameters and the master secret key which are then given to an adversary. From it, there is the gap between the real environment and the simulation environment. Therefore, to simulate a malicious KGC, we adopt the modification from [Au et al. 2007] to allow $\mathcal{A}_{II}$ to generate all the public parameters and the master secret key. In our model, an adversarial algorithm $\mathcal{A}_{II}$ runs in three stages $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2$. We define the Type II adversary's advantage as

$$\mathsf{Adv}_{\mathcal{A}_{II}^{\mathsf{CL}}}(k) = \left| \Pr\left[\mathfrak{b} = \mathfrak{b}' \middle| \begin{array}{l} (mpk, msk) \leftarrow \mathcal{A}_0(1^k), \\ (m_0, m_1, \mathsf{ID}^*, s) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathrm{II}}}(mpk, msk), \\ \mathfrak{b} \leftarrow \{0,1\}, C^* \leftarrow \mathsf{Enc}(m_{\mathfrak{b}}, pk_{\mathsf{ID}^*}, \mathsf{ID}^*, mpk), \\ \mathfrak{b}' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathrm{II}}}(C^*, s) \end{array} \right] - \frac{1}{2} \right|$$

The original model believes that the KGC honestly generates the public parameters. However, the KGC can maliciously generate them. For example, we assume that the KGC should pick three random elements (g, g_1, g_2) in a multiplicative group $\mathbb{G}$ of order p as the public parameters. The KGC maliciously generates random elements such as $g_1 = g^{\alpha}, g_2 = g^{\beta}$ from elements α, β of $\mathbb{Z}_p^*$ and it can try to break the system from α, β. Actually, we can show that there exists a CCA-secure CL-PKE scheme against Type II adversaries in previous model, but not secure against the modified Type II adversarial model [Liu et al. 2007, Dent et al. 2008]. Some previous work are broken by this attack. Therefore, we allow the type II adversary to generate all public parameters.

2.3 Assumptions

We briefly review bilinear maps and describe complexity assumptions related to our construction. Let $\mathbb{G}$ and $\mathbb{G}_T$ be the two multiplicative cyclic groups of order p for some large prime p. A *bilinear* map should satisfy the following properties:

1. Bilinear: We say that a map $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is bilinear if $e(g^a, h^b) = e(g, h)^{ab}$ for all $g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$.
2. Non-degenerate: The map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$. Observe that since $\mathbb{G}$, $\mathbb{G}$ are groups of prime order this implies that if g is a generator of $\mathbb{G}$ then $e(g, g)$ is a generator of $\mathbb{G}_T$.

3. Computable: It is efficient to compute $e(g,\, h)$ for any $g, h \in \mathbb{G}$.

The security of our construction is based on the Decisional Bilinear Diffie-Hellman (DBDH) assumption. In addition, our construction uses a collision resistant hash function to check the validity of a ciphertext.

Definition 2 Decisional Bilinear Diffie-Hellman (DBDH) Assumption. Given a group $\mathbb{G}$ of prime order p with generator g, a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and elements $g^a, g^b, g^c \in \mathbb{G}$, $e(g,g)^z \in \mathbb{G}_T$ where a, b, c, z are selected uniformly at random from $\mathbb{Z}_p^*$. A fair coin $\mathfrak{b} \in \{0,1\}$ is flipped. If $\mathfrak{b} = 1$, it outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = e(g,g)^{abc})$. If $\mathfrak{b} = 0$, it outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = e(g,g)^z)$. The problem is to guess the bit $\mathfrak{b}$.

We define the advantage of any PPT algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{\mathsf{DBDH}}(k)$, as

$$\left| \; \Pr[1 \leftarrow \mathcal{A}(g, g^a, g^b, g^c, e(g,g)^{abc})] - \Pr[1 \leftarrow \mathcal{A}(g, g^a, g^b, g^c, e(g,g)^z)] \; \right|$$

where the probability is over the randomly chosen a, b, c, z and the random bits consumed by $\mathcal{A}$. We assume that $Adv_{\mathcal{A}}^{DBDH}(k)$ is a negligible function.

Definition 3 Collision Resistant (CR) Assumption. A hash function $H \leftarrow \mathcal{H}(k)$ is collision resistant if for all PPT algorithms $\mathcal{A}$ the advantage

$$Adv_{\mathcal{A}}^{\mathsf{CR}}(k) = \Pr[H(x) = H(y) \wedge x \neq y \mid (x,y) \leftarrow \mathcal{A}(1^k, H) \wedge H \leftarrow \mathcal{H}(k)]$$

is negligible as a function of the security parameter.

Definition 4 One-Time Signature (OT). A one-time signature scheme is existential unforgeable against chosen message attack if for all PPT algorithms $\mathcal{A}$ the advantage $Adv_{\mathcal{A}}^{\mathsf{OT}}(k)$, which is

$$\Pr[\mathsf{VF}(PK, M, \sigma) = 1 \mid (PK, SK) \leftarrow \mathsf{KG}(k) \;\wedge\; (M, \sigma) \leftarrow \mathcal{A}^{\mathsf{SGN}(SK, M_q)}(PK)]$$

is negligible as a function of the security parameter, if $M \neq M_q$, where KG, SGN and VF are the key generation, signing and verification algorithm of the one-time signature scheme respectively. Note that the adversary can make at most one query to the signing oracle.

3 Malicious KGC Attack of Existing CL-PKE Schemes

In this section, we illustrate how a KGC can easily implant trapdoor in the system parameter to compromise the security of the user, using the scheme in [Liu et al. 2007] (called LAS scheme) as an example.

3.1 Review of LAS Scheme

The LAS scheme is constructed from Waters' IBE scheme [Waters 2005] by using a *message authentication code* and an *encapsulation* scheme as building blocks.

A message authentication code is a pair of PPT algorithms (Mac, Vrfy) such that Mac takes as input a key sk and a message m to produce a tag tag. The algorithm Vrfy takes as input a key sk, a message m and tag and outputs either $\top$ or $\bot$. It is required that for all sk and m, $\mathsf{Vrfy}(sk, m, \mathsf{Mac}(sk, m)) = \top$. In addition, an encapsulation scheme is a weak variant of commitment and is defined by a triple of PPT algorithms (Init, S, R) as follow. On input security parameter k', Init outputs some public parameters pub. On input k' and pub, S outputs com, dec on some appropriate range and a string $r \in \{0,1\}^{k'}$. On input pub, com and dec, R outputs r.

The LAS scheme is as follows.

- $\mathsf{Setup}(1^{k'})$. The KGC selects groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p with a pairing $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$. Let g be a generator of $\mathbb{G}$. Randomly pick $\alpha \leftarrow \mathbb{Z}_p$, $g_2 \leftarrow \mathbb{G}_1$, and compute $g_1 = g^\alpha$ and $pub = \mathsf{Init}(k')$. Also randomly select $u', g_1', h_1 \leftarrow \mathbb{G}$ and $u_i \leftarrow \mathbb{G}$ for $i = 1, \ldots, n$. Let $U = \{u_i\}$. The public parameters mpk are $(e, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, u', g_1', h_1, U, pub)$ and the master secret key msk is g_2^α.
- $\mathsf{PSK}(mpk, msk, \mathsf{ID})$. Let ID be a bit string of length n and $\mathsf{ID}[i]$ be the i-th bit. Define $\mathcal{U} \subset \{1, \ldots, n\}$ to be the set of indices i such that $\mathsf{ID}[i] = 1$. The KGC picks a random value $r \in \mathbb{Z}_p^*$ and computes;
$$psk_{\mathsf{ID}} = (psk_1 = g_2^\alpha \cdot F_u(\mathsf{ID})^r, psk_2 = g^r) \qquad \text{where } F_u(\mathsf{ID}) = u' \prod_{i\in\mathcal{U}} u_i.$$
A user with an identity ID is given psk_{ID} as a partial private key.
- $\mathsf{UKeyGen}(mpk, psk_{\mathsf{ID}})$. User selects a secret value $x \in \mathbb{Z}_p$, sets his private key $sk_{\mathsf{ID}} = (sk_1 = psk_1, sk_2 = psk_2, sk_3 = x)$ and computes his public key pk_{ID} as $(g^x, g_1^x) = (pk_1, pk_2)$.
- $\mathsf{Enc}(mpk, \mathsf{ID}, pk_{\mathsf{ID}}, m)$. To encrypt a message $m \in \{0,1\}^{\kappa'}$ ($\kappa' = \lfloor \frac{\kappa-1}{2} \rfloor$ and κ is the number of bit representing an element in $\mathbb{G}_T$) for an identity ID and public key $pk_{\mathsf{ID}} = (pk_1, pk_2)$, first check whether $pk_1, pk_2 \in \mathbb{G}$ and $e(pk_1, g_1) = e(pk_2, g)$. If not, output $\bot$ and abort encryption. Otherwise, run $\mathsf{S}(pub)$ to obtain $(r \in \{0,1\}^{k'}, com \in \mathbb{Z}_p, dec \in \{0,1\}^{\kappa'})$ and set $M = m||dec$. Randomly select $s \in_R \mathbb{Z}_p$, and compute
$$C_1 = e(pk_2, g_2)^s \cdot M \qquad C_2 = g^s \qquad C_3 = F_u(\mathsf{ID})^s \qquad C_4 = ({g_1'}^{com} h_1)^s.$$
Let $\hat{C} = (C_1, C_2, C_3, C_4)$. Compute $tag = \mathsf{Mac}(r, \hat{C})$. The ciphertext is $C = (\hat{C}, com, tag)$.
- $\mathsf{Dec}(mpk, sk_{\mathsf{ID}}, C)$. On receiving C, compute $M = C_1 \left(\frac{e(psk_2, C_3) e(g, C_4)}{e(psk_1 \cdot {g_1'}^{com} h_1, C_2)} \right)^x$ and obtain m and dec. Compute $r = \mathsf{R}(pub, com, dec)$. If $\mathsf{Vrfy}(r, \hat{C}, tag) = \top$, then the plaintext is m, else output $\bot$.

3.2 Analysis

It has shown in [Liu et al. 2007] that the security of the LAS scheme in the most common security models, where the KGC starts launching Type II attacks only after it has honestly generated the public parameters. However, the KGC can maliciously generate the parameters as follows and then decrypt every ciphertext.

- Setup. When generating the public parameters, the malicious KGC computes u' and u_i as follows:
  1. Select random values β and μ_i in $\mathbb{Z}_p$ for $i = 1, \ldots, n$.
  2. Compute $u' = g_2^{\beta}$ and $u_i = g_2^{\mu_i}$ for $i = 1, \ldots, n$.

  It then publishes $mpk = (e, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, u', g_1', h_1, U, pub)$ and securely keeps $(\beta, \mu_1, \ldots, \mu_n)$ with msk.
- Dec. Eavesdropping a ciphertext $C = (C_1, C_2, C_3, C_4, com,\ tag)$, the KGC can obtain the encoded message M by computing:
  1. $g_2^s \leftarrow (C_3)^{1/(\beta + \sum_{i \in \mathcal{U}} \mu_i)}$
     $(C_3 = F_u(\mathsf{ID})^s = (u' \prod_{i \in \mathcal{U}} u_i)^s = (g_2^{\beta + \sum_{i \in \mathcal{U}} \mu_i})^s = (g_2^s)^{\beta + \sum_{i \in \mathcal{U}} \mu_i})$
  2. $M \leftarrow C_1 / e(pk_2, g_2^s)$

The scheme in [Dent et al. 2008] shares similarities with the LAS scheme [Liu et al. 2007], thus an attack similar to the above can be applied. We remark that there is no security claim against malicious KGC attacks in [Liu et al. 2007], and it has already been stated in [Dent et al. 2008] that neither of their schemes are secure against adversaries that maliciously generate the system parameters.

4 New Construction

We construct a CCA-secure CL-PKE scheme against Type I and Type II adversaries in our modified model. Our scheme is constructed by a similar way to [Dent et al. 2008]. We apply the techniques of [Boyen et al. 2005] to the 2-level hierarchical extension of Waters' IBE [Waters 2005] to achieve the CCA-security as in [Dent et al. 2008]. To fight against the malicious KGC attack, our scheme uses a different public key generation algorithm.

- $\mathsf{Setup}(1^k)$. The KGC chooses groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p such that a pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ can be constructed and selects a generator g of $\mathbb{G}$. It picks random values $\alpha, \beta, \mu', \mu_1, \ldots, \mu_n, \nu', \nu_1, \cdots, \nu_n$ in $\mathbb{Z}_p^*$ and computes $g_1 = g^{\alpha}, h = e(g^{\alpha}, g^{\beta}), u' = g^{\mu'}, u_1 = g^{\mu_1}, \ldots, u_n = g^{\mu_n}, v' = g^{\nu'}, v_1 = g^{\nu_1}, \ldots, v_n = g^{\nu_n}$, where n is the length of an identity in binary string representation. Let $H : \{0,1\}^* \rightarrow \{0,1\}^n$ be a collision-resistant hash function. The master public key mpk and the master secret key msk are: $mpk \leftarrow (e, \mathbb{G}, \mathbb{G}_T, g, g_1, h, u', u_1, \ldots, u_n, v', v_1, \ldots, v_n, H)$ and $msk \leftarrow (\alpha, \beta, \mu', \mu_1, \ldots, \mu_n, \nu', \nu_1, \ldots, \nu_n)$ respectively.

– $\mathsf{PSK}(mpk, msk, \mathsf{ID})$. Let ID be a bit string of length n and $\mathsf{ID}[i]$ be the i-th bit. Define $\mathcal{U} \subset \{1, \ldots, n\}$ to be the set of indices i such that $\mathsf{ID}[i] = 1$. The KGC picks a random value $r \in \mathbb{Z}_p^*$ and computes;

$$psk_{\mathsf{ID}} = (psk_1, psk_2) = (g_1^{\beta} \cdot F_u(\mathsf{ID})^r, g^r) \qquad \text{where } F_u(\mathsf{ID}) = u' \prod_{i \in \mathcal{U}} u_i.$$

A user with an identity ID is given psk_{ID} as a partial private key.

– $\mathsf{UKeyGen}(mpk, psk_{\mathsf{ID}})$. Pick a secret value $x_{\mathsf{ID}} \in \mathbb{Z}_p^*$. The public key pk_{ID} is generated as $pk_{\mathsf{ID}} = (X_{\mathsf{ID}}, \sigma_{\mathsf{ID}})$ where $X_{\mathsf{ID}} = h^{x_{\mathsf{ID}}}$ and σ_{ID} is the Schnorr one-time signature using x_{ID} as the signing key and $(h, X_{\mathsf{ID}} = h^{x_{\mathsf{ID}}})$ as the verification key. The message can be any arbitrary string which can be included in mpk. The signature can be generated using the technique of Fiat-Shamir transform without random oracles as described in [Bellare and Shoup 2007]. Then it picks r' randomly from $\mathbb{Z}_p^*$ and computes the private key sk_{ID} as

$$(sk_1, sk_2) = (psk_1^{x_{\mathsf{ID}}} \cdot F_u(\mathsf{ID})^{r'}, psk_2^{x_{\mathsf{ID}}} \cdot g^{r'}) = (g_1^{\beta x_{\mathsf{ID}}} \cdot F_u(\mathsf{ID})^{r x_{\mathsf{ID}} + r'}, g^{r x_{\mathsf{ID}} + r'}).$$

– $\mathsf{Enc}(mpk, \mathsf{ID}, pk_{\mathsf{ID}}, m)$. To encrypt $m \in \mathbb{G}_T$, first check whether the public key X_{ID} is correctly formed, by checking whether σ_{ID} is a valid signature, using (h, X_{ID}) as the verification key. If not, output $\perp$ and abort the algorithm. Otherwise, select a random value $s \in \mathbb{Z}_p^*$ and compute: (Let w be a n-bit string and w_i the i-th bit of w.)

$$C = (C_0, C_1, C_2, C_3) = (m \cdot (X_{\mathsf{ID}})^s, g^s, F_u(\mathsf{ID})^s, F_v(w)^s)$$

where $w = H(C_0, C_1, C_2, \mathsf{ID}, pk_{\mathsf{ID}}) \in \{0,1\}^n$ and $F_v(w) = v' \prod_{j=1}^{n} v_j^{w_j}$.

– $\mathsf{Dec}(mpk, sk_{\mathsf{ID}}, C)$. For a ciphertext $C = (C_0, C_1, C_2, C_3)$, check that

$$e(C_1, F_u(\mathsf{ID}) \cdot F_v(w)) = e(g, C_2 C_3)$$

where $w = H(C_0, C_1, C_2, \mathsf{ID}, pk_{\mathsf{ID}}) \in \{0,1\}^n$. If not, output $\perp$. Otherwise, compute

$$m = C_0 \cdot e(C_2, sk_2)/e(C_1, sk_1).$$

Security Proof. Here, we show that our construction above is secure against a malicious KGC under the Decisional Bilinear Diffie-Hellman (DBDH) assumption in the standard model.

Theorem 5. *Let $\mathcal{A}_{II}$ be a Type II adversary that makes at most q_d decryption queries, q_{pk} public key queries, then we have*

$$Adv_{\mathcal{A}_{II}}^{\mathsf{CL}} \leq 4 q_{pk} q_d (n+1) \cdot Adv_{\mathcal{A}'}^{\mathsf{DBDH}}(k) + q_{pk} \cdot Adv_{\mathcal{A}''}^{\mathsf{CR}}(k)$$

where $\mathcal{A}'$ and $\mathcal{A}''$ are algorithms that run in approximately the same time as $\mathcal{A}_{II}$.

Proof. We define a sequence of modified attack games. Each of the games operates on the same underlying probability space. The attacker attempts to distinguish a hidden bit $\mathfrak{b}$ and eventually outputs a guess $\mathfrak{b}'$, where the hidden bit $\mathfrak{b}$ takes on identical values across all games, while some of the rules that define how a simulator responds to oracle queries may differ from game to game.

We let S_i be the event that $\mathfrak{b} = \mathfrak{b}'$ in the Game i and Adv_i denote the adversary's advantage in the Game i. Then, $Adv_i = |\Pr[S_i] - 1/2|$. We start from the Game 1 and show from the definition of Game i for $i > 1$ that $|\Pr[S_i] - 1/2|$ is negligible if and only if $|\Pr[S_{i-1}] - 1/2|$ is negligible. Let E be an event that can occur during the execution of the adversary and it is independent of S_i (i.e $\Pr[S_i|E] = \Pr[S_i]$). Let Game i+1 be the attack environment which is identical to Game i unless E occurs. If E does not occur, the adversary will choose the same bit that it did in Game i (i.e. $\Pr[S_{i+1}|\neg E] = \Pr[S_i|\neg E] = \Pr[S_i]$.) Otherwise, it outputs a random bit $\mathfrak{b}'$ (i.e. $\Pr[S_{i+1}|E] - 1/2$). Then we have

$$\begin{aligned} \big|\Pr[S_{i+1}] - 1/2\big| &= \big|\Pr[S_{i+1}|E]\Pr[E] + \Pr[S_{i+1}|\neg E]\Pr[\neg E] - 1/2\big| \\ &= \big|\Pr[E]/2 + \Pr[S_i|\neg E]\Pr[\neg E] - 1/2\big| \\ &= \big|(1 - \Pr[\neg E])/2 + \Pr[S_i]\Pr[\neg E] - 1/2\big| \\ &= \Pr[\neg E]\big|\big|\Pr[S_i] - 1/2\big|. \end{aligned}$$

Therefore, $Adv_{i+1} = \Pr[\neg E] \cdot Adv_i$.

Game 1. This game is identical to the original attack environment. A Type II adversary $\mathcal{A}_{II}$ first outputs (mpk, msk) to the simulator $\mathcal{B}$ and interacts with $\mathcal{B}$. It issues up to q_{pk}, q_{sk}, and q_d queries to $\mathsf{PKO}, \mathsf{SKO}$, and DecO respectively. We define the following sets.

- $\mathsf{pk_L} = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_{q_{pk}}\}$: the set of identities queried for public key oracle.
- $\mathsf{sk_L} = \{\mathsf{ID}'_1, \ldots, \mathsf{ID}'_{q_{sk}}\}$: the set of identities queried for private key extract oracle.
- $\mathsf{D_w} = \{w_1, \ldots, w_{q_d}\}$: the set of string $w_j = H(C_0, C_1, C_2, \mathsf{ID}_j, pk_j)$ involved in decryption queries.

$\mathcal{A}_{II}$ selects a target identity/public key pair $(\mathsf{ID}^*, pk_{\mathsf{ID}^*})$ with two equal length messages m_0, m_1, where $\mathsf{ID}^* \notin \mathsf{sk_L}$, and sends them to $\mathcal{B}$. It is given $C^* = (C_0^*, C_1^*, C_2^*, C_3^*)$ as the challenge ciphertext. At this time, we denote $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$ and $w^* \notin \mathsf{D_w}$.

Game 2. In this game, $\mathcal{B}$ first selects an identity ID_i in $\mathsf{pk_L}$ at random. Let g^a, g^b be random elements such that a, b are unknown to $\mathcal{B}$. Then, it sets X_{ID_i} by $e(g^a, g^b)^{\alpha\beta}$. At this time, x_{ID_i} is regarded as ab because $h = e(g^\alpha, g^\beta)$. π_{ID_i} can be simulated in the same way as in the signing oracle of the one-time signature. It also picks $\kappa \in \{0, \ldots, n\}$ and let τ be an integers such that $\tau(n+1) < p$. In addition, it randomly selects vectors $(x', x_1, \ldots, x_n)$ in $\mathbb{Z}_\tau$ and $(y', y_1, \ldots, y_n)$ in $\mathbb{Z}_p$ and sets:

$$v' = (g^a)^{x' - \kappa\tau} g^{y'}, \qquad v_j = (g^a)^{x_j} g^{y_j} \text{ for } 1 \leq j \leq n.$$

If $\mathcal{A}$ does not select ID_i as a target identity (namely, $\mathsf{ID}_i \neq \mathsf{ID}^*$), then $\mathcal{B}$ aborts. Therefore, $Adv_2 = \frac{1}{q_{pk}} Adv_1$.

Game 3. This game is identical to Game 2 except that $\mathcal{B}$ halts if the attacker submits a decryption query $(C, \mathsf{ID}, pk_{\mathsf{ID}})$ for a well-formed ciphertext $C = (C_0, C_1, C_2, C_3)$ where w is either equal to the same value as a previously submitted ciphertext or w is equal to w^* in the post challenge phase. For such a legal decryption query, we have $C \neq C^*$ or $(\mathsf{ID}, pk_{\mathsf{ID}}) \neq (\mathsf{ID}^*, pk_{\mathsf{ID}^*})$. In either case, this implies a collision for H. Hence, we can construct an algorithm $\mathcal{A}''$ such $|\Pr[S_2] - \Pr[S_3]| \leq Adv^{\mathsf{CR}}_{\mathcal{A}''}(k)$.

Game 4. We define the following functions from the values of Game 2 as

$$J(w) = x' + \Sigma_{j=1}^{n} w_j x_j - \kappa\tau, \qquad K(w) = y' + \Sigma_{j=1}^{n} w_j y_j$$

taking as input n-bit string $w = w_1 \ldots w_n$. Then $F_v(w) = v' \prod_{j=1}^{n} v_j^{w_j} = (g^a)^{J(w)} \cdot g^{K(w)}$.

Game 4 is the same as Game 3 except that, after $\mathcal{A}$ outputs her guess $\mathfrak{b}'$ for $\mathfrak{b}$, $\mathcal{B}$ checks whether $J(w^*) = 0 \bmod p$. If $J(w^*) \neq 0 \bmod p$, then $\mathcal{B}$ aborts and outputs a random bit $\mathfrak{b}'$. The event that $J(w^*) = 0 \bmod p$ happens by chance because $\mathcal{A}_{II}$ does not know information on all values $(x', x_1, \ldots, x_n, \kappa, \tau)$ to compute $J(w)$ at all. Actually, $\Pr[J(w^*) = 0 \bmod p] = \Pr[\kappa\tau = (x' + \Sigma_{j=1}^{n} w_j x_j)]$ since $(x' + \Sigma_{j=1}^{n} w_j x_j) < \tau(n+1), \kappa\tau < \tau(n+1)$ and $\tau(n+1) < p$. Therefore,

$$\Pr[J(w^*) = 0 \bmod p] = \frac{1}{\tau(n+1)}$$

and $Adv_4 = \frac{1}{\tau(n+1)} Adv_3$.

Game 5. We modify the way the challenge ciphertext is constructed. $\mathcal{B}$ introduces a new variable $c \leftarrow \mathbb{Z}_p^*$ and $C_1^* = g^c$. It flips a coin $\mathfrak{b}$, computes $C_0^* = m_{\mathfrak{b}} \cdot X^c_{\mathsf{ID}^*}, C_2^* = C_1^{*U_{\mathsf{ID}^*}} = (g^c)^{U_{\mathsf{ID}^*}}, C_3^* = C_1^{*K(w^*)} = (g^c)^{K(w^*)}$ where $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$ and $U_{\mathsf{ID}^*} = \sum_{i \in \mathcal{U}} \mu_j$. Clearly, $Adv_5 = Adv_4$.

Game 6. We change Game 5 so that, after $\mathcal{A}$ outputs her guess $\mathfrak{b}'$, $\mathcal{B}$ aborts and replaces $\mathcal{A}$'s output by a random bit $\mathfrak{b}'$ if $J(w_\ell) = 0 \bmod \tau$ for some $w_\ell \in \mathsf{D_w}$ where $\ell \in \{1, \ldots, q_d\}$. Since $\Pr[J(w) = 0 \bmod \tau] = 1/\tau$, $Adv_6 = (1 - \frac{1}{\tau})^{q_d} \cdot Adv_5 \geq (1 - \frac{q_d}{\tau}) \cdot Adv_5$ as $\Pr[J(w) = 0 \bmod \tau] = 1/\tau$. If we set

$\tau = 2q_d$, then $Adv_6 \geq \frac{1}{2} Adv_5$.

Game 7. We effectively change the treatment of $\mathcal{A}$'s queries. For all public key queries, private key queries and decryption key queries not involving ID^*, $\mathcal{B}$ can respond to queries by running the algorithms $\mathsf{PSK}(mpk, msk, \mathsf{ID})$, $\mathsf{UKeyGen}(mpk, psk_{\mathsf{ID}})$ and $\mathsf{Dec}(mpk, sk_{\mathsf{ID}}, C)$. In addition, it responds to all decryption queries involving ID^* as follows. When it receives decryption queries for a valid ciphertext (C_0, C_1, C_2, C_3) for ID^*, $\mathcal{B}$ aborts and outputs a random bit $\mathfrak{b}'$ as in Game 6 if $J(w) = 0 \bmod \tau$, Otherwise, $\mathcal{B}$ can extract m by computing

$$\begin{aligned}
&w \leftarrow H(C_0, C_1, C_2, \mathsf{ID}^*, pk_{\mathsf{ID}^*}) \\
&(g^a)^s \leftarrow (C_3/C_1^{K(w)})^{1/J(w)} \\
&m \leftarrow C_0/e(g^{as}, g^b)^{\alpha\beta} = C_0/e(g^\alpha, g^\beta)^{abs} = C_0/X^s_{\mathsf{ID}^*}.
\end{aligned}$$

Note that we can compute $(C_3/C_1^{K(w)})^{1/J(w)}$, since $J(w) \neq 0 \bmod p$ if $J(w) \neq 0 \bmod \tau$. We observe that $\mathcal{B}$ correctly answers $\mathcal{A}$'s queries as in Game 6. This implies $Adv_7 = Adv_6$.

Game 8. We again alter the generation of the challenge ciphertext. For a variable c introduced in Game 5, let $C_1^* = g^c$ and $Z = e(g^a, g^b)^c$. $\mathcal{B}$ retrieves values α, β, flips a coin $\mathfrak{b}$, and computes $C_0^* = m_{\mathfrak{b}} \cdot Z^{\alpha\beta}, C_2^* = (g^c)^{U_{\mathsf{ID}^*}}, C_3^* = (g^c)^{K(w^*)}$ where $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$. We have $Adv_8 = Adv_7$.

Game 9. We again alter the challenge phase. This time, $\mathcal{B}$ "forgets" the value c and simply retains C_1^*. The challenge ciphertext is constructed as in Game 8 but using a randomly chosen $Z \in \mathbb{G}_T$ this time. The whole simulation only depends on the values g^a, g^b, g^c and the simulator does not use a, b, c at all. Therefore, $|\Pr[S_8] - \Pr[S_9]| \leq Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k)$ and $\Pr[S_9] = 1/2$.

$$\begin{gathered}
Adv_6 = Adv_7 = Adv_8 \leq Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) \\
Adv_4 = Adv_5 \leq 2 \cdot Adv_6
\end{gathered}$$

Since $Adv_4 = Adv_3/(\tau(n+1))$ and $\tau = 2q_d$, we get

$$\begin{gathered}
Adv_3 \leq 4q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) \\
Adv_2 \leq Adv^{\mathsf{CR}}_{\mathcal{A}''}(k) + Adv_3 \leq 4q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) + Adv^{CR}_{\mathcal{A}''}(k)
\end{gathered}$$

In consequence, since $Adv_2 = 1/q_{pk} \cdot Adv_1$, we obtain

$$Adv_1 \leq 4q_{pk}q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) + q_{pk} \cdot Adv^{\mathsf{CR}}_{\mathcal{A}''}(k).$$

$\square$

Now we provide the security proof of our scheme against Type I adversaries.

Theorem 6. *Let $\mathcal{A}_I$ be a Type I adversary that makes at most q_d decryption queries, then we have*

$$Adv_{\mathcal{A}_I}^{\mathsf{CL}} \leq 4q_d(n+1) \cdot Adv_{\mathcal{A}'}^{\mathsf{DBDH}}(k) + Adv_{\mathcal{A}''}^{\mathsf{CR}}(k) + Adv_{\mathcal{A}_s}^{\mathsf{OT}}(k)$$

where $\mathcal{A}'$, $\mathcal{A}''$ and $\mathcal{A}_s$ are algorithms that run in approximately the same time as $\mathcal{A}_I$.

Proof. The proof is done in a way similar to that of Theorem 5.

Game 1. This game is identical to the original attack environment. $\mathcal{B}$ runs $\mathsf{Setup}(1^k)$ and outputs (mpk, msk). A Type I adversary $\mathcal{A}_I$ is given mpk. Then it issues up to q_{psk}, q_{pk}, q_{rpk}, q_{sk}, and q_d queries to PSKO, PKO, RepO, SKO, and DecO respectively. We define the following sets.

- $\mathsf{pk_L} = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_{q_{pk}}\}$: the set of identities queried for public key oracle.
- $\mathsf{rpk_L} = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_{q_{rpk}}\}$: the set of identities queried for public key replace oracle.
- $\mathsf{psk_L} = \{\mathsf{ID}'_1, \ldots, \mathsf{ID}'_{q_{psk}}\}$: the set of identities queried for partial private key extract oracle.
- $\mathsf{D_w} = \{w_1, \ldots, w_{q_d}\}$: the set of string $w_j = H(C_0, C_1, C_2, \mathsf{ID}_j, pk_j)$ involved in decryption queries.

$\mathcal{A}_I$ selects a target identity/public key pair $(\mathsf{ID}^*, pk_{\mathsf{ID}^*})$ with two equal length messages m_0, m_1, where $\mathsf{ID}^* \notin \mathsf{psk_L}$ or $\mathsf{ID}^* \notin \mathsf{rpk_L}$, and sends them to $\mathcal{B}$. It is given $C^* = (C_0^*, C_1^*, C_2^*, C_3^*)$ as the challenge ciphertext. At this time, we denote $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$ and $w^* \notin \mathsf{D_w}$.

Game 2. This game is identical to Game 1, except some value of the mpk in the system parameters are replaced with following. Let g^a be a random element in $\mathbb{G}$ such that a is unknown to $\mathcal{B}$. It randomly selects $\beta \in \mathbb{Z}_p^*$, and sets $g_1 = g^a, h = (g^a, g^\beta)$. In addition, it also picks $\kappa_v \in \{0, \ldots, n\}$ and let τ_v be an integers such that $\tau_v(n+1) < p$. It randomly selects vectors $(x'_v, x_{v,1}, \ldots, x_{v,n})$ in $\mathbb{Z}_{\tau_v}$ and $(y'_v, y_{v,1}, \ldots, y_{v,n})$ in $\mathbb{Z}_p$ and sets:

$$v' = (g^a)^{x'_v - \kappa_v \tau_v} g^{y'_v}, \qquad v_j = (g^a)^{x_{v,j}} g^{y_{v,j}} \text{ for } 1 \leq j \leq n.$$

The replaced public key has the same distribution as the public key generated in the previous game. It should also contain a valid one-time signature π. Hence, we can construct an algorithm $\mathcal{A}_s$ such that $|\Pr[S_1] - \Pr[S_2]| \leq Adv_{\mathcal{A}_s}^{\mathsf{OT}}(k)$. Note that $(\nu_1, \ldots, \nu_n)$ and α of msk (which is regarded as a) are unknown to $\mathcal{B}$. However, it securely keeps other values $(\beta, \mu', \mu_1, \ldots, \mu_n, \nu')$ of msk.

Game 3. In this game, $\mathcal{B}$ first selects an identity ID_i in $\mathsf{pk_L}$ at random. Let g^b be a random element in $\mathbb{G}$ such that b is unknown to $\mathcal{B}$. Then it sets X_{ID_i} by

$e(g^a, g^{b\beta})$. At this time, x_{ID_i} is regarded as b because $h = e(g^a, g^\beta)$. We can see that $Adv_3 = Adv_2$.

Game 4. This game is identical to Game 3 except that $\mathcal{B}$ halts if the attacker submits a decryption query $(C, \mathsf{ID}, pk_{\mathsf{ID}})$ for a well-formed ciphertext $C = (C_0, C_1, C_2, C_3)$ where w is either equal to the same value as a previously submitted ciphertext or w is equal to w^* in the post challenge phase. For such a legal decryption query, we have $C \neq C^*$ or $(\mathsf{ID}, pk_{\mathsf{ID}}) \neq (\mathsf{ID}^*, pk_{\mathsf{ID}^*})$. In either case, this implies a collision for H. Hence, we can construct an algorithm $\mathcal{A}''$ such $|\Pr[S_3] - \Pr[S_4]| \leq Adv^{\mathsf{CR}}_{\mathcal{A}''}(k)$.

Game 5. We define the following functions from the values of Game 3 as

$$J(w) = x' + \Sigma_{j=1}^{n} w_j x_j - \kappa\tau, \qquad K(w) = y' + \Sigma_{j=1}^{n} w_j y_j$$

taking as input n-bit string $w = w_1 \ldots w_n$. Then $F_v(w) = v' \prod_{j=1}^{n} v_j^{w_j} = (g^a)^{J(w)} \cdot g^{K(w)}$.

Game 5 is the same as Game 4 except that, after $\mathcal{A}$ outputs her guess $\mathfrak{b}'$ for $\mathfrak{b}$, $\mathcal{B}$ checks whether $J(w^*) = 0 \bmod p$. If $J(w^*) \neq 0 \bmod p$, then $\mathcal{B}$ aborts and outputs a random bit $\mathfrak{b}'$. The event that $J(w^*) = 0 \bmod p$ happens by chance because $\mathcal{A}_{II}$ does not know information on all values $(x', x_1, \ldots, x_n, \kappa, \tau)$ to compute $J(w)$ at all. Actually, $\Pr[J(w^*) = 0 \bmod p] = \Pr[\kappa\tau = (x' + \Sigma_{j=1}^{n} w_j x_j)]$ since $(x' + \Sigma_{j=1}^{n} w_j x_j) < \tau(n+1), \kappa\tau < \tau(n+1)$ and $\tau(n+1) < p$. Therefore,

$$\Pr[J(w^*) = 0 \bmod p] = \frac{1}{\tau(n+1)}$$

and $Adv_5 = \frac{1}{\tau(n+1)} Adv_4$.

Game 6. We modify the way the challenge ciphertext is constructed. $\mathcal{B}$ introduces a new variable $c \leftarrow \mathbb{Z}_p^*$ and $C_1^* = g^c$. It flips a coin $\mathfrak{b}$, computes $C_0^* = m_{\mathfrak{b}} \cdot X^c_{\mathsf{ID}^*}, C_2^* = C_1^{*U_{\mathsf{ID}^*}} = (g^c)^{U_{\mathsf{ID}^*}}, C_3^* = C_1^{*K(w^*)} = (g^c)^{K(w^*)}$ where $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$ and $U_{\mathsf{ID}^*} = \sum_{i \in \mathcal{U}} \mu_j$. Clearly, $Adv_6 = Adv_5$.

Game 7. We change Game 6 so that, after $\mathcal{A}$ outputs her guess $\mathfrak{b}'$, $\mathcal{B}$ aborts and replaces $\mathcal{A}$'s output by a random bit $\mathfrak{b}'$ if $J(w_\ell) = 0 \bmod \tau$ for some $w_\ell \in \mathsf{D_w}$ where $\ell \in \{1, \ldots, q_d\}$. Since $\Pr[J(w) = 0 \bmod \tau] = 1/\tau$, $Adv_6 = (1 - \frac{1}{\tau})^{q_d} \cdot Adv_5 \geq (1 - \frac{q_d}{\tau}) \cdot Adv_5$ as $\Pr[J(w) = 0 \bmod \tau] = 1/\tau$. If we set $\tau = 2q_d$, then $Adv_7 \geq \frac{1}{2} Adv_6$.

Game 8. We effectively change the treatment of $\mathcal{A}$'s queries. For all public key queries, private key queries and decryption key queries not involving ID^*, $\mathcal{B}$ can respond to queries by running the algorithms $\mathsf{PSK}(mpk, \beta, \mathsf{ID})$ (only

β in msk, instead of msk, is required to generate psk_{ID} for a queried ID), $\mathsf{UKeyGen}(mpk, psk_{\mathsf{ID}})$ and $\mathsf{Dec}(mpk, sk_{\mathsf{ID}}, C)$. When it receives a public key replace query, it replaces a previously generated public key pk_{ID} for ID with a new one pk'_{ID}. If the decryption query involves an identity ID with a replaced public key, the corresponding private key x_{ID} is required to be supplied. Otherwise it aborts. In addition, it responds to all decryption queries involving ID^* as follows. When it receives decryption queries for a valid ciphertext (C_0, C_1, C_2, C_3) for ID^*, $\mathcal{B}$ aborts and outputs a random bit $\mathfrak{b}'$ as in Game 6 if $J(w) = 0 \bmod \tau$, Otherwise, $\mathcal{B}$ can extract m by computing

$$\begin{aligned}
&w \leftarrow H(C_0, C_1, C_2, \mathsf{ID}^*, pk_{\mathsf{ID}^*})\\
&(g^a)^s \leftarrow (C_3/C_1^{K(w)})^{1/J(w)}\\
&m \leftarrow C_0/e(g^{as}, B)^\beta = C_0/e(g^a, B^\beta)^s = C_0/X^s_{\mathsf{ID}^*}.
\end{aligned}$$

where $B = g^b$ if $\mathsf{ID}^* \notin \mathsf{rpk_L}$ or $B = g^{x_{\mathsf{ID}^*}}$ otherwise, since x_{ID^*} should be given to the decryption oracle if $\mathsf{ID}^* \in \mathsf{rpk_L}$. Note that we can compute $(C_3/C_1^{K(w)})^{1/J(w)}$, since $J(w) \neq 0 \bmod p$ if $J(w) \neq 0 \bmod \tau$. We observe that $\mathcal{B}$ correctly answers $\mathcal{A}$'s queries as in Game 6. This implies $Adv_8 = Adv_7$.

Game 9. We again alter the generation of the challenge ciphertext. For a variable c introduced in Game 5, let $C_1^* = g^c$ and $Z = e(g^a, g^b)^c$. $\mathcal{B}$ retrieves values α, β, flips a coin $\mathfrak{b}$, and computes $C_0^* = m_{\mathfrak{b}} \cdot Z^\beta, C_2^* = (g^c)^{U_{\mathsf{ID}^*}}, C_3^* = (g^c)^{K(w^*)}$ where $w^* = H(C_0^*, C_1^*, C_2^*, \mathsf{ID}^*, pk_{\mathsf{ID}^*})$. We have $Adv_9 = Adv_8$.

Game 10. We again alter the challenge phase. This time, $\mathcal{B}$ "forgets" the value c and simply retains C_1^*. The challenge ciphertext is constructed as in Game 9 but using a randomly chosen $Z \in \mathbb{G}_T$ this time. The whole simulation only depends on the values g^a, g^b, g^c and the simulator does not use a, b, c at all. Therefore, $|\Pr[S_9] - \Pr[S_{10}]| \leq Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k)$ and $\Pr[S_{10}] = 1/2$.

$$\begin{gathered}
Adv_7 = Adv_8 = Adv_9 \leq Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k)\\
Adv_5 = Adv_6 \leq 2 \cdot Adv_7
\end{gathered}$$

Since $Adv_5 = Adv_4/(\tau(n+1))$ and $\tau = 2q_d$, we get

$$Adv_4 \leq 4q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k)$$

$$Adv_3 \leq Adv^{\mathsf{CR}}_{\mathcal{A}''}(k) + Adv_4 \leq 4q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) + Adv^{\mathsf{CR}}_{\mathcal{A}''}(k)$$

In consequence, since $Adv_3 = Adv_2$ and $Adv_1 \leq Adv^{\mathsf{OT}}_{\mathcal{A}_s}(k) + Adv_2$, we obtain

$$Adv_1 \leq 4q_d(n+1) \cdot Adv^{\mathsf{DBDH}}_{\mathcal{A}'}(k) + Adv^{\mathsf{CR}}_{\mathcal{A}''}(k) + Adv^{\mathsf{OT}}_{\mathcal{A}_s}(k).$$

$\square$

5 Concluding Remarks

We have showed that some previous CL-PKE scheme in the standard model are not secure against the malicious KGC attacks, and proposed a new CL-PKE scheme which is provably secure against the malicious KGC attacks in the standard model. It is believed to be the first in the literature to achieve the strongest Type II security without random oracles.

We remark that one may also construct a certificateless signature scheme secure against malicious KGC in the standard model using a similar technique presented in this paper, which will be our future work.

References

[Al-Riyami and Paterson 2003] Al-Riyami, S., and Paterson, K.: "Certificateless public key cryptography"; Proc. AsiaCrypt 2003, LNCS Vol. 2894, Springer-Verlag (2003), 452-473.

[Al-Riyami and Paterson 2005] Al-Riyami, S., and Paterson, K.: "CBE from CL-PKE: A generic construction and efficient schemes"; Proc. PKC 2005, LNCS Vol. 3386, Springer-Verlag (2005), 398-415.

[Au et al. 2007] Au, M., Chen, J., Liu, J., Mu, Y., Wong, D., and Yang, G.: "Malicious KGC Attacks in Certificateless Cryptography"; Proc. ASIACCS 2007, ACM Press (2007), 302-311.

[Bellare and Shoup 2007] Bellare, M., and Shoup, S.: "Two-tier signatures, strongly unforgeable signatures, and Fiat-Shamir without random oracles"; Proc. PKC 2007, LNCS Vol. 4450, Springer-Verlag (2007), 201-216.

[Bentahar et al. 2005] Bentahar, K., Farshim, P., Malone-Lee, J., and Smart, N. P.: "Generic construction of identity-based and certificateless KEMs"; To Appear in Journal of Cryptology, also available at also available at Cryptology ePrint Archive, Report 2005/058.

[Boyen et al. 2005] Boyen, X., Mei, Q., and Waters, B.: "Direct chosen ciphertext security from identity-based techniques"; Proc. ACMCCS 2005, ACM Press (2005), 320-329.

[Chow et al. 2006] Chow, S., Boyd, C., and Nieto, J.: "Security-Mediated Certificateless Cryptography"; Proc. PKC 2006, LNCS Vol. 3958, Springer-Verlag (2006), 508-524.

[Dent et al. 2008] Dent, A., Libert, B., and Paterson, K.: "Certificateless Encryption Schemes Strongly Secure in the Standard Model"; To appear in Proc. PKC 2008, LNCS, Springer-Verlag (2008), also available at Cryptology ePrint Archive, Report 2007/121.

[Dent 2006] Dent, A.: "A Survey of Certificateless Encryption Schemes and Security Models"; Cryptology ePrint Archive, Report 2006/211.

[Galindo et al. 2006] Galindo, D., Morillo, P., Ràfols, C.: "Breaking Yum and Lee Generic Constructions of Certificate-Less and Certificate-Based Encryption Schemes"; Proc. EuroPKI 2006, LNCS Vol. 4043, Springer-Verlag (2006), 81-91.

[Hu et al. 2006] Hu, B., Wong, D., Zhang, Z., Deng, X.: "Key replacement attack against a generic construction of certificateless signature"; Proc. ACISP 2006, LNCS Vol. 4058, Springer-Verlag (2006), 235-246.

[Libert and Quisquater 2006] Libert, B., and Quisquater, J.: "On constructing certificateless cryptosystems from identity based encryption"; Proc. PKC 2006, LNCS Vol. 3958, Springer-Verlag (2006), 474-490.

[Liu et al. 2007] Liu, J., Au, M., and Susilo, W.: "Self-Generated-Certificate Public Key Cryptography and Certificateless Signature/Encryption Scheme in the Standard Model"; Proc. ASIACCS 2007, ACM Press (2007), 273-283.

[Shamir 1984] Shamir, A.: "Identity-based Cryptosystems and Signature Schemes"; Proc. Crypto 1984, LNCS Vol. 196, Springer-Verlag (1984), 47-53.

[Waters 2005] Waters, B.: "Efficient Identity-Based Encryption Without Random Oracles"; Proc. EuroCrypt 2005, LNCS Vol. 3494, Springer-Verlag (2005), 114-127.

[Yum and Lee 2004a] Yum, D., and Lee, P.: "Generic construction of certificateless encryption"; Proc. ICCSA 2004, LNCS Vol. 3043, Springer-Verlag (2004), 802-811.

[Yum and Lee 2004b] Yum, D., and Lee, P.: "Identity-based cryptography in public key management"; Proc. EuroPKI 2004, LNCS Vol. 3093, Springer-Verlag (2004), 71-84.