Theoretical Computer Science 755 (2019) 65–80

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Efficient public key encryption with equality test in the standard model

Kai Zhang [a,b], Jie Chen [b,c], Hyung Tae Lee [d,*], Haifeng Qian [c], Huaxiong Wang [e]

[a] Department of Information Security, College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai, China
[b] State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China
[c] Department of Computer Science and Technology, East China Normal University, Shanghai, China
[d] Division of Computer Science and Engineering, College of Engineering, Chonbuk National University, South Korea
[e] Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Received 11 February 2018
Received in revised form 30 May 2018
Accepted 30 June 2018
Available online 24 August 2018
Communicated by G. Yang

Keywords:
Public key encryption with equality test
Standard model
Adaptive chosen ciphertext security

ABSTRACT

Public key encryption with equality test (PKEET) is a special kind of public encryption scheme (PKE) that allows a tester to perform equality tests on ciphertexts generated by different public keys as well as the same public key. This feature enables us to apply PKEET to various scenarios in practice, such as efficient data management on encrypted databases and spam filtering in encrypted email systems. From these reasons, since Yang et al. [1] first proposed the concept of PKEET, there have been proposed many PKEET schemes to improve efficiency or to enhance functionalities. However, to the best of our knowledge, almost all existing schemes were presented under assuming the existence of random oracles, except for generic construction proposed by Lee et al. On the other hand, their generic approach for PKEET employs a 2-level hierarchical identity-based encryption and a strongly unforgeable one-time signature, which suffers from low efficiency.

In this paper, we propose an efficient PKEET scheme under a specific cryptographic assumption in the standard model. To this end, we first encrypt a message and its hash value in a parallel way by following the recently proposed strategy. Then, to prevent adaptive chosen ciphertext attacks (CCA2), we give a link between them by adapting the technique which was originally proposed for identity-based encryption and previously exploited to design efficient CCA2-secure PKE schemes. We show that our proposed construction satisfies formal security requirements for PKEET under the decisional bilinear Diffie–Hellman (DBDH) assumption in the standard model. As a result, we obtain a new PKEET scheme which has shorter ciphertext and trapdoor sizes, and improves computational costs for encryption, decryption, and test algorithms, by about 60%, 77%, and 66%, respectively, compared to a PKEET instantiation obtained by the prior generic framework.

1. Introduction

Public key encryption with equality test (PKEET), which was first proposed by Yang et al. [1], is a special notion of public key encryption (PKE) that allows a tester to perform equality tests on encrypted data using different public keys as well as

* Corresponding author.
E-mail addresses: kzhang@shiep.edu.cn (K. Zhang), S080001@e.ntu.edu.sg (J. Chen), hyungtaelee@chonbuk.ac.kr (H.T. Lee), hfqian@cs.ecnu.edu.cn (H. Qian), hxwang@ntu.edu.sg (H. Wang).

https://doi.org/10.1016/j.tcs.2018.06.048
0304-3975/

the same public key. Specifically, suppose that U_1 and U_2 are users in a PKEET system and let CT_1 and CT_2 be ciphertexts of messages m_1 and m_2 generated by using U_1's and U_2's public keys, respectively. Some day, once a need arises, each user U_i issues a trapdoor td_i to a tester. Since then, the tester who has trapdoors, can perform equality tests between CT_1 and CT_2 using td_1 and td_2, to check whether m_1 is equal to m_2 or not. This property enables PKEET schemes to be applied for various scenarios in practice, e.g., keyword search on encrypted data, efficient data management on encrypted databases, spam filtering in encrypted email systems, and personal health record systems.

Since being introduced in [1], due to its broad applications, there have been proposed many PKEET schemes to enhance functionalities and/or to improve efficiency. However, almost all existing PKEET constructions [1–7] were given in the random oracle model [8], which generally views exploited hash functions as random oracles in the security analysis. However, random oracles do not exist in reality and so a PKEET construction that was proven secure in the random oracle model may turn into be insecure when random oracles are instantiated with truly hash functions, as in other cryptographic primitives [9].

To the best of our knowledge, there is only one PKEET construction in the standard model. Very recently, Lee et al. [10] presented a generic construction for PKEET derived from a 2-level hierarchical identity-based encryption (HIBE) scheme which is secure under selective identity and chosen plaintext attacks and a strongly unforgeable one-time signature scheme. This approach immediately yields the first specific PKEET construction in the standard model by employing Boneh and Boyen's 2-level HIBE scheme [11] and Boneh, Shen, and Waters' strongly unforgeable signature [12]. However, those cause inefficiency in both computational costs and parameter sizes.

1.1. Our results

In this paper, we propose a new PKEET construction in the standard model. Our proposed scheme achieves security requirements for PKEET schemes under assuming the decisional bilinear Diffie–Hellman (DBDH) assumption holds and exploited hash functions are one-way and collision resistant. Differently from Lee et al.'s generic approach [10], our construction is designed directly without employing strong cryptographic primitives, such as HIBE schemes and strongly unforgeable signatures. As a result, our proposed scheme outperforms the specific scheme obtained from Lee et al.'s generic approach by exploiting Boneh and Boyen's 2-level HIBE scheme [11] and Boneh, Shen, and Waters' strongly unforgeable signature scheme [12], in terms of both parameter sizes and computational costs.

Strategy for our PKEET construction. Similarly to recently proposed PKEET constructions [7,10], we begin by generating two ciphertexts of a message and its hash value under different public keys, respectively. Let $\mathbb{G}, \mathbb{G}_T$ be multiplicative groups of prime order p and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. Let g be a random generator of $\mathbb{G}$. Consider an ElGamal-type encryption

$$\mathsf{E}(\mathrm{PK}, m) = (m \cdot e(g^{\alpha}, h)^{s}, g^{s})$$

where a public key PK is $(g, e(g^{\alpha}, h))$ for randomly chosen $\alpha \in \mathbb{Z}_p$ and $h \in \mathbb{G}$ in the key generation algorithm and s is the randomness chosen in the encryption algorithm. Then, we consider our provisional encryption algorithm $\overline{\mathsf{E}}$ for message m as

$$\begin{aligned}\overline{\mathsf{E}}(\overline{\mathrm{PK}}, m) &= (\mathsf{E}(\mathrm{PK}_1, m), \mathsf{E}(\mathrm{PK}_2, \mathsf{H}_1(m)) \\ &= ((m \cdot e(g^{\alpha}, h)^{s}, g^{s}), (\mathsf{H}_1(m) \cdot e(g^{\beta}, h)^{s'}, g^{s'})) \end{aligned} \tag{1}$$

where a public key $\overline{\mathrm{PK}} = (\mathrm{PK}_1, \mathrm{PK}_2) = ((g, e(g^{\alpha}, h)), (g, e(g^{\beta}, h)))$ for randomly chosen $\alpha, \beta \in \mathbb{Z}_p$ and $h \in \mathbb{G}$ in the key generation algorithm, s, s' are randomness chosen by the encryption algorithm, and $\mathsf{H}_1 : \mathbb{G}_T \to \mathbb{G}_T$ is a cryptographic hash function.

Next, to maximize the efficiency of our construction, we modify our encryption algorithm so that both ciphertexts of the encryption algorithm E share the same randomness. Thus, we have a modified encryption algorithm

$$\mathsf{E}'(\overline{\mathrm{PK}}, m) = (m \cdot e(g^{\alpha}, h)^{s}, \mathsf{H}_1(m) \cdot e(g^{\beta}, h)^{s}, g^{s}).$$

However, unfortunately, a ciphertext obtained by the above encryption algorithm E' can still be manipulated easily by an adversary who can perform adaptive chosen ciphertext attacks (CCA2), similarly to the CCA2 attack for the ElGamal encryption [13]. (We remark that the ElGamal encryption does not achieve CCA2 security.)

To prevent CCA2 attacks, we adapt the technique presented by Lai et al. [14], which was originally proposed to obtain an efficient CCA2-secure PKE scheme in the standard model. In [14], the authors paid attention that many (generic) CCA2-secure PKE schemes in the standard model were constructed using identity-based encryption and then attempted to apply an identity-based technique to design a direct CCA2-secure PKE scheme. As a result, they obtained an efficient CCA2-secure PKE scheme under the DBDH assumption by attaching two additional components to a ciphertext of an ElGamal-type PKE, where it requires only 3 additional exponentiations to compute them. By adapting their technique, we finally obtain our encryption algorithm as follows:

$$\begin{aligned}\mathsf{Enc}(\mathrm{PK}, m) &= (C_0, C_1, C_2, C_3, C_4)\\ &= \left(m \cdot e(g^{\alpha}, h)^{s},\ \mathsf{H}_1(m) \cdot e(g^{\beta}, h)^{s},\ g^{s}, (u^{t} v^{r} \omega)^{s}, r\right) \end{aligned} \tag{2}$$

where a public key $\mathrm{PK} = (g, e(g^{\alpha}, h), e(g^{\beta}, h), u, v, \omega)$ for randomly chosen $u, v, \omega \in \mathbb{G}$ in the key generation algorithm, $r \in \mathbb{Z}_p$ is a random element chosen by the encryption algorithm, $t = \mathsf{H}_2(C_0, C_1, C_2)$, and $\mathsf{H}_2 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G} \to \mathbb{Z}_p$ is a cryptographic hash function.

Informally, in the CCA2 security game, the adversary who wants to ask a decryption query by manipulating the challenge ciphertext, cannot obtain a valid modification of the challenge ciphertext because she does not know the exact randomness s utilized in the challenge ciphertext and thus she cannot generate C_3 in Equation (2) for another valid ciphertext correctly. On the other hand, if a tester has the value h^{β}, which is a secret key corresponding to $\mathsf{E}(\mathrm{PK}_2, \mathsf{H}_1(m))$ in Equation (1), then he can obtain $\mathsf{H}_1(m)$ from Equation (2) since it holds that

$$C_1 / e(C_2, h^{\beta}) = \mathsf{H}_1(m) \cdot e(g^{\beta}, h)^{s} / e(g^{s}, h^{\beta}) = \mathsf{H}_1(m).$$

Thus, he can perform equality tests by comparing $\mathsf{H}_1(m)$ values. In this case, we expect that a tester cannot obtain the exact value m from given $\mathsf{H}_1(m)$ if the exploited hash function H_1 is one-way. (To this end, as in other PKEET schemes, we also assume that the size of message space is exponential in λ and the min-entropy of the message distribution is sufficiently higher than λ for security parameter λ. See Remark 1 for details.)

We formally show that our construction achieves one-wayness under adaptive chosen ciphertext attacks (OW-CCA2) against Type-I adversaries who have trapdoors for equality tests and indistinguishability under adaptive chosen ciphertext attacks (IND-CCA2) against Type-II adversaries who do not have trapdoors. They are achieved in the standard model under assuming that the DBDH assumption holds and exploited hash functions are one-way and collision-resistant.

Our proposed scheme has better performance than the outcome obtained from Lee et al.'s generic construction [10] by employing Boneh and Boyen's 2-level HIBE scheme [11] and Boneh, Shen, and Waters' signature scheme [12]: Our construction reduces the number of group elements in a ciphertext from linear in λ to constant for security parameter λ. The computational costs for encryption, decryption, and test algorithms are improved by about 60%, 77%, and 66%, respectively. Furthermore, parameter sizes and the efficiency of our encryption algorithm are also comparable to those of the existing efficient schemes [4,6] in the random oracle model. (Refer to Table 2 in Section 5 for the details of efficiency comparison.)

1.2. Related work

In this subsection, we briefly review existing PKEET constructions.

PKEET in the random oracle model. Since Yang et al. [1] introduced the concept of PKEET and presented the first instantiation, there have been proposed various PKEET constructions in the random oracle model. Tang [2,3] proposed a security-enhanced PKEET scheme in the sense that the system model separates a tester who has trapdoors for equality tests and others who do not have. However, his constructions exploited an interactive protocol between a receiver and a tester to issue a trapdoor and it caused a drawback of efficiency. Later, Tang [4] proposed an efficiency-improved version by removing an interactive protocol. We note that our scheme and several recently proposed schemes [6,7,10] follow this system model. Ma et al. [5] gave a PKE scheme with delegated equality test that employs an entity that is the only party who can access to storage. Independently, Huang et al. [15] proposed a PKE scheme with authorized equality test where it can issue two types of trapdoors for a specified ciphertext and all user's ciphertexts, respectively. However, their scheme was not IND-CCA2 secure and Lee et al. [16] recently pointed out and rectified it. Ma et al. [6] proposed a PKEET construction that supports four types of authorizations. Very recently, Lee et al. [7] provided a semi-generic construction for PKEET. We say that their scheme is semi-generic in the sense that a specific scheme can be obtained by employing IND-CCA2 secure PKE schemes, and one-way and collision-resistant hash functions with the computational Diffie–Hellman assumption.

PKEET in the standard model. As far as we know, there exists only one PKEET construction in the standard model. Very recently, Lee et al. [10] proposed a generic construction for PKEET which was derived from a 2-level HIBE scheme that is secure under selective identity and chosen plaintext attacks and a strongly unforgeable one-time signature. Let us elaborate their construction: Assume that there are a 2-level HIBE scheme $\mathsf{HIBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and a one-time signature $\mathsf{OTS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. Then, a ciphertext of their scheme for message m consists of $\mathrm{CT} = (\mathrm{VK}, C_0, C_1, C_2)$ where VK is a verification key generated by executing the key generation algorithm of one-time signature $\mathsf{OTS.KeyGen}(\lambda)$ for input a security parameter λ,

$$C_0 = \mathsf{HIBE.Enc}([0.\mathrm{VK}], m),\ C_1 = \mathsf{HIBE.Enc}([1.\mathrm{VK}], \mathsf{H}_1(m)),\ \text{and } C_2 = \mathsf{OTS.Sign}(\mathrm{SK}, [C_0 \| C_1]).$$

Here, SK is a signing key corresponding to VK, H_1 is a cryptographic hash function, and $[\mathsf{ID}_1.\mathsf{ID}_2]$ denotes an identity whose i-th level identity is ID_i for $i = 1, 2$. In this scheme, the secret key SK_1 for the first-level identity 1 is a trapdoor since a tester who has SK_1 can generate a secret key of identity $[1.\mathrm{VK}]$ for any VK using SK_1 and thus he can perform equality tests by recovering $\mathsf{H}_1(m)$ from C_1. Their scheme yields the first specific construction by employing existing HIBE schemes

and one-time signatures in the standard model. However, unfortunately, there are no efficient HIBE scheme and one-time signature in the standard model, which generate PKEET schemes that have comparable performance to existing PKEET schemes in the random oracle model.

Roadmap. In Section 2, we introduce the system model and definitions for our PKEET construction and give preliminaries used in our construction. We present our PKEET scheme in Section 3 and analyze its security in Section 4. Section 5 compares features and efficiency of our work with related works and Section 6 concludes this work.

2. Preliminaries

In this section, we introduce our PKEET system model and provide formal definitions for PKEET and its security model. Then, we recall cryptographic assumptions and primitives that will be utilized in our construction.

Notations. For a finite set S, we denote by $s \xleftarrow{\$} S$ the process of uniformly sampling a random element s from S. If A is an algorithm, $a \leftarrow A$ denotes that A outputs a. Denote by $[n]$ the set $\{1, \ldots, n\}$. We say that a function $f : \mathbb{N} \to \mathbb{R}$ is negligible if $f(\lambda) \leq 1/p(\lambda)$ for all polynomials $p(\cdot)$ and sufficiently large λ.

2.1. Public key encryption with equality test

System model for our PKEET. Our PKEET system consists of the following three entities, a sender(s), a receiver(s), and a tester(s): When a sender wants to send his/her data to a receiver, he/she encrypts his/her data using the receiver's public key and sends a ciphertext to the receiver. The receiver may decrypt the ciphertext using his/her secret key and/or store it at the server. Once a need arises, the receiver issues a trapdoor for equality tests of all his/her ciphertexts to a tester who can access to the server. Since then, the tester can perform equality tests on the ciphertexts encrypted using the public key of the receiver who passed the trapdoor to the tester.

Before introducing definitions of PKEET, we remark that a PKEET system is formulated in a multi-user setting. Throughout the paper, we assume that each user is assigned an index i for $1 \leq i \leq N$ where N is the number of users in the system. For notational convenience, we often use a subscripted index to represent a user and his/her ciphertexts, keys and trapdoors. For example, U_i is the i-th user and PK_i, CT_i, and td_i are a public key, a ciphertext, and a trapdoor for user U_i, respectively.

Definition 1. A public key encryption with equality test (PKEET) consists of the following six polynomial-time algorithms:

- $\mathsf{Setup}(\lambda)$: On input a security parameter λ, it outputs the public parameter PP.

  We note that PP includes $\mathcal{M}$ where $\mathcal{M}$ denotes the message space of the PKEET scheme. We also remark that all the following algorithms take PP as an input, although it is not explicitly stated.
- $\mathsf{KeyGen}(\mathrm{PP})$: It takes the public parameter PP as an input and outputs a pair of user's public and secret keys $(\mathrm{PK}, \mathrm{SK})$.
- $\mathsf{Enc}(\mathrm{PK}, m)$: It takes the public key PK and a message m as inputs and outputs a ciphertext CT.
- $\mathsf{Dec}(\mathrm{SK}, \mathrm{CT})$: On input the secret key SK and a ciphertext CT, it outputs a message m' or $\perp$.
- $\mathsf{Td}(\mathrm{SK}_i)$: It takes the secret key SK_i for user U_i as an input and returns a trapdoor td_i.
- $\mathsf{Test}(\mathrm{td}_i, \mathrm{td}_j, \mathrm{CT}_i, \mathrm{CT}_j)$: It takes two trapdoors $\mathrm{td}_i, \mathrm{td}_j$ and two ciphertexts $\mathrm{CT}_i, \mathrm{CT}_j$ for users U_i, U_j, respectively, as inputs and outputs 1 or 0.

Correctness. We say that a PKEET scheme is *correct* if the following three conditions hold:
(1) For any security parameter λ, any user U_i and any message $m \in \mathcal{M}$, it holds that

$$\Pr\left[\mathsf{Dec}(\mathrm{SK}_i, \mathrm{CT}_i) = m \;\middle|\; \begin{array}{l} \mathrm{PP} \leftarrow \mathsf{Setup}(\lambda); \\ (\mathrm{PK}_i, \mathrm{SK}_i) \leftarrow \mathsf{KeyGen}(\mathrm{PP}); \\ \mathrm{CT}_i \leftarrow \mathsf{Enc}(\mathrm{PK}_i, m) \end{array}\right] = 1.$$

(2) For any security parameter λ, any users U_i, U_j and any messages $m_i, m_j \in \mathcal{M}$, it holds that

$$\Pr\left[\mathsf{Test}\begin{pmatrix} \mathrm{td}_i, \\ \mathrm{td}_j, \\ \mathrm{CT}_i, \\ \mathrm{CT}_j \end{pmatrix} = 1 \;\middle|\; \begin{array}{l} \mathrm{PP} \leftarrow \mathsf{Setup}(\lambda); \\ (\mathrm{PK}_i, \mathrm{SK}_i) \leftarrow \mathsf{KeyGen}(\mathrm{PP}); \\ \mathrm{CT}_i \leftarrow \mathsf{Enc}(\mathrm{PK}_i, m_i); \\ \mathrm{td}_i \leftarrow \mathsf{Td}(\mathrm{SK}_i); \\ (\mathrm{PK}_j, \mathrm{SK}_j) \leftarrow \mathsf{KeyGen}(\mathrm{PP}); \\ \mathrm{CT}_j \leftarrow \mathsf{Enc}(\mathrm{PK}_j, m_j); \\ \mathrm{td}_j \leftarrow \mathsf{Td}(\mathrm{SK}_j) \end{array}\right] = 1$$

if $m_i = m_j$ regardless of whether $i = j$ or not.

(3) For any security parameter λ, any users U_i, U_j and any two messages $m_i, m_j \in \mathcal{M}$, it holds that

$$\Pr\left[\mathsf{Test}\begin{pmatrix}\mathsf{td}_i,\\ \mathsf{td}_j,\\ \mathsf{CT}_i,\\ \mathsf{CT}_j\end{pmatrix}=1 \;\middle|\; \begin{array}{l}\mathsf{PP}\leftarrow\mathsf{Setup}(\lambda);\\ (\mathsf{PK}_i,\mathsf{SK}_i)\leftarrow\mathsf{KeyGen}(\mathsf{PP});\\ \mathsf{CT}_i\leftarrow\mathsf{Enc}(\mathsf{PK}_i,m_i);\\ \mathsf{td}_i\leftarrow\mathsf{Td}(\mathsf{SK}_i);\\ (\mathsf{PK}_j,\mathsf{SK}_j)\leftarrow\mathsf{KeyGen}(\mathsf{PP});\\ \mathsf{CT}_j\leftarrow\mathsf{Enc}(\mathsf{PK}_j,m_j);\\ \mathsf{td}_j\leftarrow\mathsf{Td}(\mathsf{SK}_j)\end{array}\right]$$

is negligible in λ for any two ciphertexts CT_i and CT_j such that $\mathsf{Dec}(\mathsf{SK}_i,\mathsf{CT}_i)\neq\mathsf{Dec}(\mathsf{SK}_j,\mathsf{CT}_j)$ regardless of whether $i=j$ or not.

2.2. Security model for PKEET

Since a tester can have trapdoors for equality tests, we consider two types of adversaries for the security of PKEET schemes, depending on whether an adversary has a trapdoor for the target user or not. If the adversary has a trapdoor for the target user, then she can perform equality tests on the challenge ciphertext. Thus, we cannot expect to achieve the indistinguishability-based security notion against her. Instead, one-wayness of the target ciphertext is expected. On the other hand, if the adversary cannot have a trapdoor, we can expect to achieve the indistinguishability-based security notion, as in traditional PKE schemes. In summary, we consider the following two types of adversaries:

- Type-I adversary: This type of adversaries can request to issue a trapdoor for the target user and thus can perform equality tests on the challenge ciphertext. Hence, we regard that the aim of this type of adversaries is to reveal the message in the challenge ciphertext.
- Type-II adversary: This type of adversaries cannot request to issue a trapdoor for the target user and thus cannot perform equality tests on the challenge ciphertext. Hence, we regard that the aim of this type of adversaries is to distinguish which message is in the challenge ciphertext between two candidates.

OW-CCA2 security against Type-I adversaries. We define an experiment played by a challenger and a Type-I adversary $\mathcal{A}$ who *can* have a trapdoor for all ciphertexts of the target user U_θ as follows:

$$\begin{array}{l}\mathsf{Experiment}\ \mathsf{Exp}^{\mathsf{OW\text{-}CCA2}}_{\mathcal{A},\mathsf{PKEET}}(\lambda)\\ \quad \mathsf{PP}\leftarrow\mathsf{Setup}(\lambda);\ U_\theta\leftarrow\mathcal{A}(\mathsf{PP});\\ \quad (\mathsf{PK}_i,\mathsf{SK}_i)\leftarrow\mathsf{KeyGen}(\mathsf{PP})\ \text{for}\ 1\le i\le N;\\ \quad \mathsf{req}\leftarrow\mathcal{A}^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot),\mathcal{O}^{\mathsf{Td}}(\cdot)}(\{\mathsf{PK}_i\}_{i=1}^N);\\ \quad m\xleftarrow{\$}\mathsf{Sample}(\mathsf{req});\ \mathsf{CT}^*_\theta\leftarrow\mathsf{Enc}(\mathsf{PK}_\theta,m);\\ \quad m'\leftarrow\mathcal{A}^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot),\mathcal{O}^{\mathsf{Td}}(\cdot)}(\{\mathsf{PK}_i\}_{i=1}^N,\mathsf{CT}^*_\theta)\end{array}$$

where $m\xleftarrow{\$}\mathsf{Sample}(\mathsf{req})$ denotes that the challenger samples a random message m from $\mathcal{M}$ once it receives the challenge request req from $\mathcal{A}$. Here, $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$ is a secret key extraction oracle that takes an index i as an input and returns a secret key of user U_i, $\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot)$ is a decryption oracle that takes an index i and a ciphertext CT_i for user U_i as inputs and returns a message m_i which is the outcome of $\mathsf{Dec}(\mathsf{SK}_i,\mathsf{CT}_i)$, and $\mathcal{O}^{\mathsf{Td}}(\cdot)$ is a trapdoor extraction oracle that takes an index i as an input and returns a trapdoor td_i for user U_i. In the above experiment, there are two restrictions for $\mathcal{A}$ that (1) the index of the target user θ cannot be queried to the secret key extraction oracle $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$ and (2) the pair of the index of the target user and the challenge ciphertext $(\theta,\mathsf{CT}^*_\theta)$ cannot be queried to the decryption oracle $\mathcal{O}^{\mathsf{Dec}}(\cdot)$.

A PKEET scheme is *OW-CCA2 secure against Type-I adversaries* if for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, its advantage function defined as

$$\mathsf{Adv}^{\mathsf{OW\text{-}CCA2}}_{\mathcal{A},\mathsf{PKEET}}(\lambda):=\Pr[m'=m]$$

in the above experiment is negligible in λ.

Remark 1. We remark that a Type-I adversary can perform equality tests between the challenge ciphertext and ciphertexts of any messages, generated by herself, since she has the trapdoor for the challenge ciphertext. Thus, if the size of message space is polynomial in the security parameter λ or the min-entropy of the message distribution is not as high as λ, the adversary can reveal the message in the challenge ciphertext within polynomial time or sufficiently small exponential time in λ. Therefore, we assume that the size of message space is exponential in λ and the min-entropy of the message distribution is sufficiently higher than λ.

IND-CCA2 security against Type-II adversaries. We define an experiment played by a challenger and a Type-II adversary $\mathcal{A}$ who *cannot* have a trapdoor for all ciphertexts for the target user U_θ as follows:

$$
\begin{aligned}
&\textsf{Experiment } \mathsf{Exp}^{\text{IND-CCA2}}_{\mathcal{A},\text{PKEET}}(\lambda)\\
&\quad \text{PP} \leftarrow \mathsf{Setup}(\lambda);\ U_\theta \leftarrow \mathcal{A}(\text{PP});\\
&\quad (\text{PK}_i, \text{SK}_i) \leftarrow \mathsf{KeyGen}(1^\lambda) \text{ for } 1 \le i \le N;\\
&\quad (m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot),\mathcal{O}^{\mathsf{Td}}(\cdot)}(\{\text{PK}_i\}_{i=1}^N);\\
&\quad \gamma \xleftarrow{\$} \{0,1\};\ \text{CT}^*_{\theta,\gamma} \leftarrow \mathsf{Enc}(\text{PK}_\theta, m_\gamma);\\
&\quad \gamma' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot),\mathcal{O}^{\mathsf{Td}}(\cdot)}(\{\text{PK}_i\}_{i=1}^N, \text{CT}^*_{\theta,\gamma})
\end{aligned}
$$

where $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$, $\mathcal{O}^{\mathsf{Dec}}(\cdot,\cdot)$, and $\mathcal{O}^{\mathsf{Td}}(\cdot)$ are oracles defined as the same as in the experiment for the OW-CCA2 security of PKEET. Here, there are two restrictions for $\mathcal{A}$ that (1) the index for the target user θ cannot be queried to the trapdoor generation oracle $\mathcal{O}^{\mathsf{Td}}(\cdot)$ as well as the secret key generation oracle $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$, and (2) the pair of the index of the target user and the challenge ciphertext $(\theta, \text{CT}^*_{\theta,\gamma})$ cannot be queried to the decryption oracle $\mathcal{O}^{\mathsf{Dec}}(\cdot)$.

A PKEET scheme is *IND-CCA2 secure against Type-II adversaries*, if for any PPT adversary $\mathcal{A}$, its advantage function defined as

$$\mathsf{Adv}^{\text{IND-CCA2}}_{\mathcal{A},\text{PKEET}}(\lambda) := \left|\Pr[\gamma' = \gamma] - \frac{1}{2}\right|$$

in the above experiment is negligible in λ.

2.3. Cryptographic assumptions

Now, we review cryptographic assumptions that we will utilize in our construction.

Decisional bilinear Diffie–Hellman assumption. The security of our construction relies on the decisional bilinear Diffie–Hellman (DBDH) assumption. Consider the following game between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$: Let Gen be an algorithm that takes a security parameter λ as an input and outputs a pair $(p, \mathbb{G}, \mathbb{G}_T, e, g)$ where $\mathbb{G}$, $\mathbb{G}_T$ are multiplicative cyclic groups of order p, e is a bilinear map from $\mathbb{G} \times \mathbb{G}$ to $\mathbb{G}_T$, and g is a generator of $\mathbb{G}$. $\mathcal{C}$ first obtains $(p, \mathbb{G}, \mathbb{G}_T, e, g)$ by executing $\mathsf{Gen}(1^\lambda)$. $\mathcal{C}$ then chooses random elements a, b, c from $\mathbb{Z}_p$ and tosses an unbiased random coin β. If $\beta = 1$, $\mathcal{C}$ sets T to $e(g,g)^{abc}$. Otherwise (i.e., $\beta = 0$), $\mathcal{C}$ sets T to a randomly chosen element from $\mathbb{G}_T$. $\mathcal{C}$ passes the instance (g, g^a, g^b, g^c, T) to $\mathcal{A}$. $\mathcal{A}$ then outputs a guess $\beta' \in \{0,1\}$. We define an advantage of $\mathcal{A}$ in the above game as $\left|\Pr[\beta' = \beta] - \frac{1}{2}\right|$.

Definition 2. We say that the decisional bilinear Diffie–Hellman assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if for any adversary, its advantage in the above game is negligible in the security parameter λ.

Properties of hash functions. We will exploit one-wayness and collision resistance of hash functions for the correctness and the security of our construction. We provide formal definitions of those properties below.

Definition 3. We say that a function $H: X \to Y$ is one-way if H can be computed by a polynomial time algorithm, but any PPT adversary $\mathcal{A}$ that attempts to recover the pre-image for H succeeds with negligible probability, that is, for $y \xleftarrow{\$} Y$

$$\Pr[\mathsf{H}(x) = y \mid x \leftarrow \mathcal{A}(\lambda, \mathsf{H}, y)]$$

is negligible in λ.

Definition 4. We say that a function $H: X \to Y$ is collision-resistant if H can be computed by a polynomial time algorithm, but any PPT adversary $\mathcal{A}$ that finds a collision for H with negligible probability, that is,

$$\Pr[(x' \neq x) \wedge (\mathsf{H}(x) = \mathsf{H}(x')) \mid x, x' \leftarrow \mathcal{A}(\lambda, \mathsf{H})]$$

is negligible in λ.

3. Our PKEET construction

In this section, we present our PKEET construction and look into the correctness of our scheme. First, the description of our PKEET scheme is as follows.

Setup(λ): Given a security parameter λ, generate a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ where $\mathbb{G}$, $\mathbb{G}_T$ are two cyclic groups of prime order $p = p(\lambda)$. Choose a random generator g of group $\mathbb{G}$. Set $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g)$. Generate two cryptographic hash functions $\mathsf{H}_1 : \mathbb{G}_T \to \mathbb{G}_T$ and $\mathsf{H}_2 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G} \to \mathbb{Z}_p$. Output a system parameter

$$\text{PP} = (\mathcal{G}, \mathsf{H}_1, \mathsf{H}_2).$$

KeyGen(PP): Given the public parameter PP, select random elements α, β, x, y, z from $\mathbb{Z}_p$. Set $g_1 = g^\alpha$, $g_2 = g^\beta$, $u = g^x$, $v = g^y$, $\omega = g^z$. Pick a random element $h \xleftarrow{\$} \mathbb{G}$ and output a public key and a secret key

$$\text{PK} = (A := e(g_1, h), B := e(g_2, h), u, v, \omega),$$
$$\text{SK} = (K_1, K_2, K_3, K_4, K_5) = (h^\alpha, h^\beta, x, y, z).$$

Enc(PK, m): On input PK and a message $m \in \mathbb{G}_T$, it randomly chooses $s, r \xleftarrow{\$} \mathbb{Z}_p$, computes

$$C_0 = m \cdot A^s, C_1 = \mathsf{H}_1(m) \cdot B^s, C_2 = g^s, C_3 = (u^t v^r \omega)^s$$

where $t = \mathsf{H}_2(C_0, C_1, C_2)$, and sets $C_4 = r$. Then, it outputs a ciphertext $\text{CT} = (C_0, C_1, C_2, C_3, C_4)$.

Dec(SK, CT): On input $\text{SK} = (K_1, K_2, K_3, K_4, K_5)$ and $\text{CT} = (C_0, C_1, C_2, C_3, C_4)$, it performs as follows:

1. Compute $t = \mathsf{H}_2(C_0, C_1, C_2)$.
2. Check whether $(C_2)^{tK_3 + C_4K_4 + K_5} = C_3$. If it does not hold, output $\perp$.
3. Otherwise, compute

$$\mathsf{H}' = \frac{C_1}{e(C_2, K_2)} \text{ and } m' = \frac{C_0}{e(C_2, K_1)}.$$

4. Output m' if $\mathsf{H}' = \mathsf{H}_1(m')$ and 0 otherwise.

Td(SK_i): Given a user U_i's secret key $\text{SK}_i = (K_{i,1}, K_{i,2}, K_{i,3}, K_{i,4}, K_{i,5})$, it outputs a trapdoor $\text{td}_i = K_{i,2}$.

Test($\text{td}_i, \text{td}_j, \text{CT}_i, \text{CT}_j$): It takes trapdoors td_i, td_j and ciphertexts CT_i, CT_j for users U_i, U_j, respectively, as inputs. Then, it parses $\text{CT}_i = (C_{i,0}, C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4})$ and $\text{CT}_j = (C_{j,0}, C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4})$, and computes

$$\mathsf{H}'_i = \frac{C_{i,1}}{e(C_{i,2}, \text{td}_i)} \text{ and } \mathsf{H}'_j = \frac{C_{j,1}}{e(C_{j,2}, \text{td}_j)}.$$

Output 1 if $\mathsf{H}'_i = \mathsf{H}'_j$ and 0 otherwise.

Remark 2. To improve the efficiency of the encryption process, we avoid pairing operations in computing (C_0, C_1) of ciphertexts by handing them over to the key generation algorithm. From our current construction, one may consider to define our scheme on a group over a finite field, not a pairing group: For example, let $\mathbb{G}$ be a multiplicative subgroup of $\mathbb{Z}_q^*$ for sufficiently large prime q and g be a generator of $\mathbb{G}$ whose order is prime p. Set $A = g^\alpha$ and $B = g^\beta$ in the public key where α and β are secret keys randomly chosen from $\mathbb{Z}_p$. The trapdoor is β. We carefully guess this provisional construction works correctly.

However, unfortunately, we cannot find an appropriate way to implement decryption oracle queries in the security game under that setting. Thus, to adapt Lai et al.'s technique [14] which was originally presented to construct IND-CCA2 PKE schemes, our scheme is defined on pairing groups. (See the proofs of Theorems 2 and 3 for decryption oracle queries.)

Correctness. The following theorem demonstrates the correctness of our PKEET construction.

Theorem 1. *Our PKEET construction described in the above is correct if* H_1 *and* H_2 *are collision-resistant hash functions.*

Proof. 1. Let $\text{CT} = (C_0, C_1, C_2, C_3, C_4)$ be a valid ciphertext of message m under the public key PK. Then, for $\text{SK} = (K_1, K_2, K_3, K_4, K_5) = (h^\alpha, h^\beta, x, y, z)$, we have

$$(C_2)^{tK_3 + C_4K_4 + K_5} = (C_2)^{tx + ry + z} = (g^s)^{tx + ry + z} = (u^t v^r \omega)^s = C_3$$

where $t = \mathsf{H}_2(C_0, C_1, C_2)$ and $r = C_4$ since $u = g^x$, $v = g^y$, and $\omega = g^z$. Furthermore, we obtain

$$\frac{C_0}{e(C_2, K_1)} = \frac{m \cdot e(g_1, h)^s}{e(g^s, h^\alpha)} = \frac{m \cdot e(g^\alpha, h)^s}{e(g, h)^{\alpha s}} = m = m'$$

and

$$\frac{C_1}{e(C_2, K_2)} = \frac{\mathsf{H}_1(m) \cdot e(g_2, h)^s}{e(g^s, h^\beta)} = \frac{\mathsf{H}_1(m) \cdot e(g^\beta, h)^s}{e(g, h)^{\beta s}} = \mathsf{H}_1(m) = \mathsf{H}'. \tag{3}$$

Thus, it holds $\mathsf{H}' = \mathsf{H}_1(m')$ and the decryption algorithm always outputs m.

2. Let $\mathrm{CT}_k := (C_{k,0}, C_{k,1}, C_{k,2}, C_{k,3}, C_{k,4})$ be a valid ciphertext of message m_k under the user U_k's public key PK_k for $k = i, j$. If $m_i = m_j$, then $\mathsf{H}'_i = \mathsf{H}_1(m_i) = \mathsf{H}_1(m_j) = \mathsf{H}'_j$ since for $k = i, j$

$$\frac{C_{k,1}}{e(C_{k,2}, K_{k,2})} = \mathsf{H}_1(m_k) = \mathsf{H}'_k$$

from Equation (3) where the user U_k's secret key is $\mathrm{SK}_k = (K_{k,1}, K_{k,2}, K_{k,3}, K_{k,4}, K_{k,5})$. Thus, the Test algorithm always outputs 1.
3. Otherwise (i.e., if $m_i \neq m_j$), $\mathsf{H}'_i = \mathsf{H}_1(m_i) \neq \mathsf{H}_1(m_j) = \mathsf{H}'_j$ with overwhelming probability since H_1 is collision-resistant. Thus, the Test algorithm outputs 1 with negligible probability.

From the above, the proof of Theorem 1 is completed. □

4. Security analysis of our PKEET construction

In this section, we show that our PKEET scheme is OW-CCA2 secure against Type-I adversaries and IND-CCA2 secure against Type-II adversaries. Recall that a Type-I adversary can have a trapdoor for the target user's ciphertexts and then can perform an equality test with the challenge ciphertext CT^*_θ, whereas a Type-II adversary cannot have a trapdoor for the target user's ciphertexts and thus cannot perform an equality test with CT^*_θ.

First, we look into OW-CCA2 security of our PKEET construction against Type-I adversaries.

Theorem 2. *Assume that the DBDH assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$, H_1 *is a one-way hash function, and* H_2 *is a collision-resistant hash function. Then, our proposed PKEET scheme in Section 3 is* OW-CCA2 secure against Type-I adversaries *in the standard model.*

Proof. Assume that there exists a PPT Type-I adversary $\mathcal{A}$ who breaks OW-CCA2 security of our PKEET construction with non-negligible probability $\varepsilon_{\mathrm{OURS}}$. Then, using $\mathcal{A}$, we construct a simulator $\mathcal{B}$ who solves the DBDH problem on instance (g, g^a, g^b, g^c, T) where T is the solution of the DBDH problem, $e(g,g)^{abc}$, or a random element in $\mathbb{G}_T$. Let N be the number of users in our PKEET system and U_θ be the target user in the OW-CCA2 security game for our construction where θ is chosen by $\mathcal{A}$ from the set $[N]$ once she receives the public parameter from $\mathcal{B}$. Denote by $\mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ the challenge ciphertext in the security game.

Description of simulator $\mathcal{B}$'s behaviors. First, we describe $\mathcal{B}$'s behaviors below.

1. Once the DBDH instance (g, g^a, g^b, g^c, T) with $(p, \mathbb{G}, \mathbb{G}_T, e)$ is given by the challenger $\mathcal{C}$ of the game for solving the DBDH problem, $\mathcal{B}$ first generates two cryptographic hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{Z}_p$ and $\mathsf{H}_2 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G} \to \mathbb{Z}_p$, sets a system parameter $\mathrm{PP} = (\mathcal{G} := (p, \mathbb{G}, \mathbb{G}_T, e, g), \mathsf{H}_1, \mathsf{H}_2)$, and sends PP to $\mathcal{A}$. $\mathcal{A}$ returns an index θ for the target user. $\mathcal{B}$ then executes $(\mathrm{PK}_i, \mathrm{SK}_i) \leftarrow \mathsf{KeyGen}(\mathrm{PP})$ for $1 \leq i \neq \theta \leq N$. For PK_θ, $\mathcal{B}$ picks random elements $\eta, x_v, x_\omega, y_u, y_v, y_\omega \in \mathbb{Z}_p$ and sets

$$g_1 = g^a,\ g_2 = g^\eta,\ h = g^b,\ u = g^b g^{y_u} = g^{b+y_u},\ v = (g^b)^{x_v} g^{y_v} = g^{bx_v + y_v},\ \omega = (g^b)^{x_\omega} g^{y_\omega} = g^{bx_\omega + y_\omega}.$$

Then it sets $\mathrm{PK}_\theta := (e(g_1, h), e(g_2, h), u, v, \omega)$. We note that the secret key SK_θ for user U_θ satisfies the following relation

$$\mathrm{SK}_\theta := (h^\alpha = h^a = g^{ab},\ h^\beta = (g^b)^\eta = g^{b\eta},\ x = b + y_u,\ y = bx_v + y_v,\ z = bx_\omega + y_\omega),$$

although $\mathcal{B}$ cannot have h^α, x, y, and z explicitly since $\mathcal{B}$ does not know a and b. Finally, $\mathcal{B}$ passes all PK_i's to $\mathcal{A}$.
2. For $\mathcal{A}$'s queries, $\mathcal{B}$ responds as follows. First, since $\mathcal{B}$ has all secret keys SK_i's for $1 \leq i \neq \theta \leq N$, $\mathcal{B}$ can respond correctly to all $\mathcal{O}^{\mathsf{KeyGen}}(i)$, $\mathcal{O}^{\mathsf{Td}}(i)$, and $\mathcal{O}^{\mathsf{Dec}}(i, \mathrm{CT}_i)$ queries using SK_i if $i \neq \theta$. $\mathcal{A}$ cannot request a query on θ to $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$ and $\mathcal{B}$ can respond correctly to $\mathcal{O}^{\mathsf{Td}}(\theta)$ by sending $h^\beta = (g^b)^\eta$ because η was chosen by $\mathcal{B}$. Lastly, for $\mathcal{A}$'s decryption queries on $(\theta, \mathrm{CT}_\theta)$ where $\mathrm{CT}_\theta = (C_{\theta,0}, C_{\theta,1}, C_{\theta,2}, C_{\theta,3}, C_{\theta,4})$, $\mathcal{B}$ performs as follows: $\mathcal{B}$ computes $t_\theta = \mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C_{\theta,2})$ and checks whether

$$e(C_{\theta,2}, u^{t_\theta} v^{r_\theta} \omega) = e(g, C_{\theta,3}) \tag{4}$$

where $r_\theta = C_{\theta,4}$. If it does not hold, output $\perp$. We note that $C_{\theta,2} = g^{s_\theta}$ and $C_{\theta,3} = (u^{t_\theta} v^{r_\theta} \omega)^{s_\theta}$ for some $s_\theta \in \mathbb{Z}_p$ if the submitted ciphertext is valid, and thus it holds Equation (4). Next, it checks whether

$$t_\theta + r_\theta x_v + x_\omega = 0. \tag{5}$$

If it holds, $\mathcal{B}$ aborts and returns a random guess. We denote the event that submitted ciphertexts hold Equation (5) by Event_1. Otherwise, $\mathcal{B}$ computes $\mathsf{H}' = C_{\theta,1} / e(C_{\theta,2}, h^\eta)$ using the knowledge of η. Since

$$\begin{aligned}\mathsf{H}' &= C_{\theta,1}/e(C_{\theta,2}, h^{\eta}) \\ &= \mathsf{H}_1(m_\theta)\cdot e(g^{\eta}, g^{b})^{s_\theta}/e(g^{s_\theta}, (g^{b})^{\eta}) \\ &= \mathsf{H}_1(m_\theta)\end{aligned}$$

where $s_\theta \in \mathbb{Z}_p$ is the randomness in CT_θ, it holds that $\mathsf{H}' = \mathsf{H}_1(m_\theta)$ if the given ciphertext is valid and m_θ is the message in the ciphertext CT_θ. Next, $\mathcal{B}$ randomly chooses $s'_\theta \in \mathbb{Z}_p$, computes

$$\begin{aligned}K_{\theta,1,1} &= g_1^{-\frac{t_\theta y_u + r_\theta y_v + y_\omega}{t_\theta + r_\theta x_v + x_\omega}} (u^{t_\theta} v^{r_\theta} w)^{s'_\theta}, \\ K_{\theta,1,2} &= g_1^{-\frac{1}{t_\theta + r_\theta x_v + x_\omega}} g^{s'_\theta}\end{aligned}$$

and then computes

$$m' = C_{\theta,0}\cdot\frac{e(C_{\theta,3}, K_{\theta,1,2})}{e(C_{\theta,2}, K_{\theta,1,1})}.$$

Since $C_{\theta,3} = (u^{t_\theta} v^{r_\theta}\omega)^{s_\theta}$ if it is valid, we have

$$\begin{aligned}e(C_{\theta,3}, K_{\theta,1,2})/e(C_{\theta,2}, K_{\theta,1,1}) &= \frac{e((u^{t_\theta} v^{r_\theta}\omega)^{s_\theta}, g_1^{-\frac{1}{t_\theta + r_\theta x_v + x_\omega}} g^{s'_\theta})}{e(g^{s_\theta}, g_1^{-\frac{t_\theta y_u + r_\theta y_v + y_\omega}{t_\theta + r_\theta x_v + x_\omega}} (u^{t_\theta} v^{r_\theta}\omega)^{s'_\theta})} \\ &= \frac{e((g^{t_\theta(b+y_u)+r_\theta(bx_v+y_v)+bx_\omega+y_\omega})^{s_\theta}, g_1^{-\frac{1}{t_\theta + r_\theta x_v + x_\omega}})}{e(g^{s_\theta}, g_1^{-\frac{t_\theta y_u + r_\theta y_v + y_\omega}{t_\theta + r_\theta x_v + x_\omega}})} \\ &= e((g^{bt_\theta + br_\theta x_v + bx_\omega})^{s_\theta}, g_1^{-\frac{1}{t_\theta + r_\theta x_v + x_\omega}}) \\ &= e((g^{b})^{s_\theta}, g^{-a}) \\ &= e(g, g)^{-abs_\theta}\end{aligned} \tag{6}$$

and thus

$$\begin{aligned}m' &= C_{\theta,0}\cdot\frac{e(C_{\theta,3}, K_{\theta,1,2})}{e(C_{\theta,2}, K_{\theta,1,1})} \\ &= m_\theta\cdot e(g_1, h)^{s_\theta}\cdot e(g, g)^{-abs_\theta} \\ &= m_\theta\cdot e(g^{a}, g^{b})^{s_\theta}\cdot e(g, g)^{-abs_\theta} = m_\theta\end{aligned}$$

if the given ciphertext is valid and m_θ is the message in the submitted ciphertext CT_θ. Finally, $\mathcal{B}$ checks whether $\mathsf{H}' = \mathsf{H}_1(m')$. If it holds, $\mathcal{B}$ outputs m' for $\mathcal{A}$'s decryption query. Otherwise, it returns $\perp$.

3. Once $\mathcal{A}$ returns the challenge request req, $\mathcal{B}$ picks a random message m from $\mathcal{M}$, computes

$$\begin{aligned}C^*_{\theta,0} &= m\cdot T, \\ C^*_{\theta,1} &= \mathsf{H}_1(m)\cdot e(g^{c}, (g^{b})^{\eta}) = \mathsf{H}_1(m)\cdot e(g, g)^{\eta bc},\end{aligned}$$

where (g, g^a, g^b, g^c, T) is the challenge of the DBDH problem, and sets

$$C^*_{\theta,2} = g^{c}.$$

Then, it computes $t^*_\theta = \mathsf{H}_2(C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2})$ and

$$C^*_{\theta,4} = r^*_\theta = -\frac{t^*_\theta + x_\omega}{x_v}.$$

Finally, it computes

$$C^*_{\theta,3} = (g^{c})^{t^*_\theta y_u + r^*_\theta y_v + y_\omega}$$

and then returns

$$\mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$$

as the challenge ciphertext. We note that $C^*_{\theta,0}$ is the first part of the valid ciphertext if T is the solution of the DBDH problem. Otherwise, $C^*_{\theta,0}$ is independent of m. Since g^c, x_v, and x_ω firstly appear in the challenge ciphertext, $C^*_{\theta,2}$ and $C^*_{\theta,4}$ look random in the view of $\mathcal{A}$. $C^*_{\theta,1}$ and $C^*_{\theta,3}$ are also valid since

$$C^*_{\theta,1} = \mathsf{H}_1(m) \cdot e(g_2, h)^c = \mathsf{H}_1(m) \cdot e(g, g)^{\eta bc}$$

and

$$\begin{aligned} C^*_{\theta,3} &= (g^c)^{t^*_\theta y_u + r^*_\theta y_v + y_\omega} \\ &= (g^c)^{t^*_\theta y_u - \frac{t^*_\theta + x_\omega}{x_v} y_v + y_\omega} \\ &= (g^{t^*_\theta (b+y_u) - \frac{(t^*_\theta + x_\omega)}{x_v}(bx_v + y_v) + (bx_\omega + y_\omega)})^c \\ &= ((g^{b+y_u})^{t^*_\theta} \cdot (g^{bx_v + y_v})^{r^*} \cdot g^{bx_\omega + y_\omega})^c \\ &= (u^{t^*_\theta} v^{r*} \omega)^c. \end{aligned}$$

4. $\mathcal{B}$ continues to respond to $\mathcal{A}$'s queries as almost the same as in the previous, except for the followings:
   (a) If $\mathcal{A}$ submits a decryption query on the challenge ciphertext CT^*_θ, $\mathcal{B}$ returns $\perp$.
   (b) Else if the submitted ciphertext CT_θ is of the form

$$\mathrm{CT}_\theta = (C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4}) \tag{7}$$

   such that $\mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}) = t^*_\theta$, but $\mathrm{CT}_\theta \neq \mathrm{CT}^*_\theta$, then $\mathcal{B}$ aborts and returns a random guess. We denote this event by Event_2.
   (c) Otherwise, it checks whether

$$t_\theta + r_\theta x_v + x_\omega = 0 \tag{8}$$

   where $t_\theta = \mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C_{\theta,2})$ and $r_\theta = C_{\theta,4}$. If the above holds, $\mathcal{B}$ aborts and returns a random guess. We denote this event by Event_3.
5. Finally, once $\mathcal{A}$ outputs m', $\mathcal{B}$ outputs 1 if $m' = m$. Otherwise, it returns $\perp$.

Analysis of $\mathcal{B}$'s advantage. Now, we analyze $\mathcal{B}$'s advantage. We first note that any abortion in the experiment does not leak any meaningful information for $\mathcal{A}$. Particularly, when Event_3 occurs, it may leak some information of x_v and x_ω. However, Equation (8) can hold for p possible pairs of (x_v, x_ω) and each of them is satisfied equally. Thus, information-theoretically, the probability that each pair is actual value (x_v, x_ω) used to generate (v, ω) is at most $1/p$. Furthermore, the information of x_v and x_ω in v and ω is perfectly hidden by y_v and y_ω, respectively. Finally, Equation (4) gives the information that $C_{\theta,2}$ and $C_{\theta,3}$ share the same randomness only.

Next, we consider the following cases that $\mathcal{B}$'s simulation fails.

(1) $\mathsf{Fail}_{\mathsf{H}_1}$: Since $\mathcal{A}$ has the trapdoor $h^\beta = h^\eta = (g^b)^\eta$ for user U_θ, $\mathcal{A}$ may obtain

$$\mathsf{H}_1(m) = C_{\theta,1} / e(C_{\theta,2}, h^\eta)$$

from $C_{\theta,1}$ and $C_{\theta,2}$. Thus, $\mathcal{A}$ may obtain any information of the challenge message m from $\mathsf{H}_1(m)$, not $C^*_{\theta,0}$. In fact, we cannot expect $\mathcal{A}$'s behaviors after $\mathsf{H}_1(m)$ is given to $\mathcal{A}$, but we can estimate the bound of the probability that $\mathcal{A}$ obtains the correct m from $\mathsf{H}_1(m)$ by constructing a simulator $\overline{\mathcal{B}}$ that breaks the one-wayness of H_1.

To construct $\overline{\mathcal{B}}$ using $\mathcal{A}$, for given the instance H', when $\overline{\mathcal{B}}$ generates the challenge ciphertext, $\overline{\mathcal{B}}$ embeds H' into the challenge ciphertext as follows: First, select random elements $s, r \xleftarrow{\$} \mathbb{Z}_p$ and a random message $m \xleftarrow{\$} \mathcal{M}$. Then, compute

$$C_{\theta,0} = m \cdot e(g_1, h)^s,\ C_{\theta,1} = \mathsf{H}' \cdot e(g_2, h)^s,\ C_{\theta,2} = g^s,\ C_{\theta,3} = (u^t v^r \omega)^s,\ C_{\theta,4} = r$$

where $t = \mathsf{H}_2(C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2})$ and send $\mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ to $\mathcal{A}$. Then, $\mathcal{A}$ finally returns the pre-image of H' and so $\overline{\mathcal{B}}$ can break the one-wayness of H_1 using $\mathcal{A}$'s answer. Thus, we have

$$\Pr[\mathsf{Fail}_{\mathsf{H}_1}] \leq \varepsilon_{\mathsf{H}_1,\mathsf{OW}}$$

where $\varepsilon_{\mathsf{H}_1,\mathsf{OW}}$ is the success probability that $\mathcal{A}$ breaks the one-wayness of H_1.

(2) $\mathsf{Abort}_{\mathcal{B}}$: Let $\mathsf{Abort}_{\mathcal{B}}$ denote the union of events that $\mathcal{B}$ aborts in the simulation.

- Event_1 and Event_3: Information-theoretically, it tells that the probability that the equation $t_\theta + r_\theta x_v + x_\omega = 0$ holds is at most $1/p$ where r_θ is randomly chosen from $\mathbb{Z}_p$, t_θ is independent of r_θ, and x_v and x_ω are fixed. Thus, we have

$$\Pr[\mathsf{Event}_1 \vee \mathsf{Event}_3] \leq \Pr[\mathsf{Event}_1] + \Pr[\mathsf{Event}_3] \leq q \cdot 1/p$$

  where $\mathcal{A}$ allows at most q queries.

- Event_2: This event occurs when $\mathcal{A}$ generates a ciphertext $\mathrm{CT}_\theta = (C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ such that $\mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}) = t^*_\theta$, but $\mathrm{CT}_\theta \neq \mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$. On the other hand, it holds that $\mathsf{H}_2(C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}) = t^*_\theta$ and thus we find a collision $(C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2})$, $(C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2})$ of the hash function H_2. Hence, we have

$$\Pr[\mathsf{Event}_2] \leq \varepsilon_{\mathsf{H}_2,\mathsf{CR}}$$

where $\varepsilon_{\mathsf{H}_2,\mathsf{CR}}$ is the success probability that $\mathcal{A}$ breaks the collision resistance of H_2.

From the above, we have

$$\Pr[\mathsf{Abort}_{\mathcal{B}}] = \Pr[\mathsf{Event}_1 \vee \mathsf{Event}_2 \vee \mathsf{Event}_3] \leq q/p + \varepsilon_{\mathsf{H}_2,\mathsf{CR}}.$$

Furthermore, let $\mathcal{F}$ be the union of the events $\mathsf{Fail}_{\mathsf{H}_1}$ and $\mathsf{Abort}_{\mathcal{B}}$. Then, we have

$$\Pr[\mathcal{F}] = \Pr[\mathsf{Fail}_{\mathsf{H}_1} \vee \mathsf{Abort}_{\mathcal{B}}] \leq \varepsilon_{\mathsf{H}_1,\mathsf{OW}} + \varepsilon_{\mathsf{H}_2,\mathsf{CR}} + q/p.$$

Now, we are ready to compute the advantage of $\mathcal{B}$. Denote by $\mathsf{Succ}_{\mathcal{B}}$ the event that $\mathcal{B}$ outputs the correct answer in the DBDH game. Recall that the DBDH challenge instance is (g, g^a, g^b, g^c, T) where $T = T_1 = (g,g)^{abc}$ or $T = T_0 \xleftarrow{\$} \mathbb{G}_T$. Then, we have

$$\begin{aligned}
\varepsilon_{\mathcal{B},\mathsf{DBDH}} &= \Pr[\mathsf{Succ}_{\mathcal{B}} \wedge \neg\mathcal{F}] \\
&\geq \Pr[\mathsf{Succ}_{\mathcal{B}} \wedge \neg\mathcal{F} \wedge (m' = m)] \\
&= \frac{1}{2} \cdot (\Pr[\mathsf{Succ}_{\mathcal{B}} \wedge \neg\mathcal{F} \wedge (m' = m) \mid T = T_1] + \Pr[\mathsf{Succ}_{\mathcal{B}} \wedge \neg\mathcal{F} \wedge (m' = m) \mid T = T_0]) \\
&\geq \frac{1}{2} \cdot \Pr[\mathsf{Succ}_{\mathcal{B}} \mid \neg\mathcal{F} \wedge (m' = m) \wedge (T = T_1)] \cdot \Pr[\neg\mathcal{F} \wedge (m' = m) \mid T = T_1] + 0 \\
&= \frac{1}{2} \cdot 1 \cdot \Pr[\neg\mathcal{F} \wedge (m' = m) \mid T = T_1] \\
&= \frac{1}{2} \cdot \Pr[m' = m \mid \neg\mathcal{F} \wedge (T = T_1)] \cdot \Pr[\neg\mathcal{F} \mid T = T_1] \\
&= \frac{1}{2} \cdot \mathsf{Adv}^{\mathsf{OW-CCA}}_{\mathcal{A}} \cdot (1 - \Pr[\mathcal{F} \mid T = T_1]) \\
&= \frac{1}{2} \cdot \varepsilon_{\mathrm{OURS}} \cdot (1 - \Pr[\mathcal{F} \wedge (T = T_1)] \cdot \frac{1}{2}) \\
&\geq \frac{1}{4} \cdot \varepsilon_{\mathrm{OURS}} \cdot (2 - \Pr[\mathcal{F}]) \\
&\geq \frac{1}{4} \cdot (2 - \varepsilon_{\mathsf{H}_1,\mathsf{OW}} - \varepsilon_{\mathsf{H}_2,\mathsf{CR}} - q/p) \cdot \varepsilon_{\mathrm{OURS}}.
\end{aligned}$$

The third and eighth steps hold since $\Pr[T = T_0] = \Pr[T = T_1] = 1/2$. The fifth step holds because when $\mathcal{B}$'s simulation does not fail and $T = e(g,g)^{abc}$ along with $\mathcal{A}$ outputting $m' = m$, the probability that $\mathcal{B}$ outputs the correct answer is 1. The seventh step holds because when $\mathcal{B}$'s simulation does not fail and $T = e(g,g)^{abc}$, $\mathcal{A}$'s view is identical to its view in the real security game and thus the probability that $\mathcal{A}$ outputs $m' = m$ is the same as the advantage of $\mathcal{A}$.

Therefore, we obtain the relation

$$\varepsilon_{\mathrm{OURS}} \leq \frac{4}{2 - \varepsilon_{\mathsf{H}_1,\mathsf{OW}} - \varepsilon_{\mathsf{H}_2,\mathsf{CR}} - q/p} \cdot \varepsilon_{\mathcal{B},\mathsf{DBDH}}. \quad \square$$

Now, we look into IND-CCA2 security of our PKEET construction against Type-II adversaries.

Theorem 3. *Assume that the DBDH assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$ *and* H_2 *is a collision-resistant hash function. Then, our proposed PKEET scheme in Section 3 is* IND-CCA2 secure against Type-II adversaries *in the standard model.*

Proof. We note that the proof of this theorem is similar to the proof of Theorem 2. Assume that there exists a PPT Type-II adversary $\mathcal{A}$ who breaks IND-CCA2 security of our PKEET construction with non-negligible probability $\varepsilon_{\mathrm{OURS}}$. Then, we construct a simulator $\mathcal{B}$ who solves the DBDH problem on instance (g, g^a, g^b, g^c, T) using $\mathcal{A}$, where T is the solution of the DBDH problem, $e(g,g)^{abc}$, or a random element in $\mathbb{G}_T$. Assume that there are N users in our PKEET system.

Description of simulator $\mathcal{B}$'s behaviors. We first describe $\mathcal{B}$'s behaviors.

1. When the DBDH instance (g, g^a, g^b, g^c, T) with $(p, \mathbb{G}, \mathbb{G}_T, e)$ is given by the challenger $\mathcal{C}$ of the game for solving the DBDH problem, $\mathcal{B}$ generates two cryptographic hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{Z}_p$ and $\mathsf{H}_2 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G} \to \mathbb{Z}_p$, sets a system parameter $\mathrm{PP} = (\mathcal{G} := (p, \mathbb{G}, \mathbb{G}_T, e, g), \mathsf{H}_1, \mathsf{H}_2)$, and forwards PP to $\mathcal{A}$. $\mathcal{A}$ submits an index θ for the target user to $\mathcal{B}$. Then, $\mathcal{B}$ runs $(\mathrm{PK}_i, \mathrm{SK}_i) \leftarrow \mathsf{KeyGen}(\mathrm{PP})$ for $1 \le i \ne \theta \le N$. To generate PK_θ, $\mathcal{B}$ first tosses an unbiased coin $\delta \in \{0,1\}$ and selects random elements $\eta, x_v, x_\omega, y_u, y_v, y_\omega \in \mathbb{Z}_p$. According to the value δ, $\mathcal{B}$ behaves differently:
(1) If $\delta = 0$, $\mathcal{B}$ sets $\mathrm{PK}_\theta = (e(g_1, h), e(g_2, h), u, v, \omega)$ where

$$\boxed{g_1 = g^a, g_2 = g^\eta}, h = g^b,$$

$u = g^b g^{y_u}$, $v = (g^b)^{x_v} g^{y_v}$, and $\omega = (g^b)^{x_\omega} g^{y_\omega}$, and regards that the secret key is

$$\mathrm{SK}_\theta := \left(h^\alpha = h^a = g^{ab},\ h^\beta = h^\eta = g^{b\eta},\ x = b + y_u,\ y = bx_v + y_v,\ z = bx_\omega + y_\omega\right).$$

We note that $\mathcal{B}$ cannot have h^α.
(2) If $\delta = 1$, $\mathcal{B}$ sets $\mathrm{PK}_\theta = (e(g_1, h), e(g_2, h), u, v, \omega)$ where

$$\boxed{g_1 = g^\eta, g_2 = g^\alpha}, h = g^b,$$

$u = g^b g^{y_u}$, $v = (g^b)^{x_v} g^{y_v}$, and $\omega = (g^b)^{x_\omega} g^{y_\omega}$, and regards that the secret key is

$$\mathrm{SK}_\theta := \left(h^\alpha = h^\eta = g^{\eta b},\ h^\beta = h^a = g^{ab},\ x = b + y_u,\ y = bx_v + y_v,\ z = bx_\omega + y_\omega\right).$$

We note that $\mathcal{B}$ cannot have h^β.
Finally, $\mathcal{B}$ forwards all PK_i's for $1 \le i \le N$ to $\mathcal{A}$.
2. For $\mathcal{A}$'s queries, $\mathcal{B}$ responds differently depending on the choice of δ. We note that $\mathcal{O}^{\mathsf{Td}}(\theta)$ query is not allowed to $\mathcal{A}$ in this experiment.
(1) If $\delta = 0$, $\mathcal{B}$'s responses are exactly the same as those in the proof of Theorem 2.
(2) If $\delta = 1$, $\mathcal{B}$ can respond correctly to $\mathcal{O}^{\mathsf{KeyGen}}(i)$, $\mathcal{O}^{\mathsf{Td}}(i)$ and $\mathcal{O}^{\mathsf{Dec}}(i, \mathrm{CT}_i)$ queries using the secret key SK_i for $1 \le i \ne \theta \le N$. For decryption queries on $(\theta, \mathrm{CT}_\theta)$ where $\mathrm{CT}_\theta = (C_{\theta,0}, C_{\theta,1}, C_{\theta,2}, C_{\theta,3}, C_{\theta,4})$, $\mathcal{B}$ performs as follows: $\mathcal{B}$ first computes $t_\theta = \mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C_{\theta,2})$ and checks if

$$e(C_{\theta,2}, u^{t_\theta} v^{r_\theta} \omega) = e(g, C_{\theta,3})$$

where $r_\theta = C_{\theta,4}$. If it does not hold, $\mathcal{B}$ outputs $\perp$. Otherwise, it checks if

$$t_\theta + r_\theta x_v + x_\omega = 0. \tag{9}$$

If it holds, $\mathcal{B}$ aborts and returns a random guess. Otherwise, $\mathcal{B}$ randomly chooses $s'_\theta \in \mathbb{Z}_p$ and computes

$$K_{\theta,2,1} = g_1^{-\frac{t_\theta y_u + r_\theta y_v + y_\omega}{t_\theta + r_\theta x_v + x_\omega}} (u^{t_\theta} v^{r_\theta} w)^{s'_\theta},$$
$$K_{\theta,2,2} = g_1^{-\frac{1}{t_\theta + r_\theta x_v + x_\omega}} g^{s'_\theta}$$

and then computes

$$\mathsf{H}' = C_{\theta,1} \cdot \frac{e(C_{\theta,3}, K_{\theta,2,2})}{e(C_{\theta,2}, K_{\theta,2,1})}.$$

Since $C_{\theta,3} = (u^{t_\theta} v^{r_\theta} \omega)^{s_\theta}$ for the randomness s_θ in CT_θ if it is valid, we obtain $e(C_{\theta,3}, K_{\theta,2,2})/e(C_{\theta,2}, K_{\theta,2,1}) = e(g,g)^{-abs_\theta}$ as Equation (6) and thus

$$\begin{aligned}\mathsf{H}' &= C_{\theta,0} \cdot \frac{e(C_{\theta,3}, K_{\theta,2,2})}{e(C_{\theta,2}, K_{\theta,2,1})} \\ &= \mathsf{H}_1(m_\theta) \cdot e(g_2, h)^{s_\theta} \cdot e(g,g)^{-abs_\theta} \\ &= \mathsf{H}_1(m_\theta) \cdot e(g^a, g^b)^{s_\theta} \cdot e(g,g)^{-abs_\theta} = \mathsf{H}_1(m_\theta)\end{aligned}$$

if the given ciphertext is valid and m_θ is the message in the submitted ciphertext CT_θ. Next, $\mathcal{B}$ computes $m' = C_{\theta,0}/e(C_{\theta,2}, h^\eta)$ by directly using the knowledge of η ($h^\eta = h^\alpha$). Since

$$\begin{aligned}m' &= C_{\theta,0}/e(C_{\theta,2}, h^\eta) \\ &= m_\theta \cdot e(g^\eta, g^b)^{s_\theta} / e(g^{s_\theta}, (g^b)^\eta) = m_\theta\end{aligned}$$

if the given ciphertext is valid and m_θ is the message in the ciphertext CT_θ, then $\mathcal{B}$ has $m' = m_\theta$. Finally, $\mathcal{B}$ checks whether $\mathsf{H}' = \mathsf{H}_1(m')$. If it does not hold, $\mathcal{B}$ outputs $\perp$. Otherwise, $\mathcal{B}$ outputs m' for $\mathcal{A}$'s decryption query.

3. Upon receiving two messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}$ picks a random bit $\gamma \in \{0,1\}$ and computes the challenge ciphertext differently depending on the value δ: For the DBDH instance (g, g^a, g^b, g^c, T), set $C^*_{\theta,2} = g^c$. Then,
(1) if $\delta = 0$, compute

$$\boxed{C^*_{\theta,0} = m_\gamma \cdot T,\ C^*_{\theta,1} = \mathsf{H}_1(m_\gamma) \cdot e(C^*_{\theta,2}, (g^b)^\eta),}$$

(2) if $\delta = 1$, compute

$$\boxed{C^*_{\theta,0} = m_\gamma \cdot e(C^*_{\theta,2}, (g^b)^\eta),\ C^*_{\theta,1} = \mathsf{H}_1(m_\gamma) \cdot T.}$$

Then, compute $t^*_\theta = \mathsf{H}_2(C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2})$, set $C^*_{\theta,4} = r^*_\theta = -(t^*_\theta + x_\omega)/x_v$ and compute

$$C^*_{\theta,3} = (g^c)^{t^*_\theta y_u + r^*_\theta y_v + y_\omega}.$$

Finally, $\mathcal{B}$ passes $\mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ to $\mathcal{A}$ as the challenge ciphertext. We note that we confirm the validity of the challenge ciphertext as in the proof of Theorem 2.

4. $\mathcal{B}$ continues to respond to $\mathcal{A}$'s queries as the same as Step 2, except for the cases defined at Step 4 of $\mathcal{B}$'s behaviors in the proof of Theorem 2. For those cases, $\mathcal{B}$ performs as at Step 4 of $\mathcal{B}$'s behaviors in the proof of Theorem 2.
5. Lastly, $\mathcal{A}$ outputs its answer γ'. If $\gamma' = \gamma$, $\mathcal{B}$ outputs 1.

Analysis of $\mathcal{B}$'s advantage. Now, we evaluate $\mathcal{B}$'s advantage. By the same reason in the proof of Theorem 2, we first note that any abortion does not leak any meaningful information for $\mathcal{A}$. We also consider the cases that $\mathcal{B}$ aborts in the experiment below.

- Event_1 and Event_3: These events are the cases that the submitted ciphertext satisfies Equation (9) at Step 2 and Step 4, respectively. From the argument in the proof of Theorem 2, we know that

$$\Pr[\mathsf{Event}_1 \vee \mathsf{Event}_3] \le \Pr[\mathsf{Event}_1] + \Pr[\mathsf{Event}_3] \le q \cdot 1/p$$

where $\mathcal{A}$ allows at most q queries.
- Event_2: This event is the case that the submitted ciphertext is of the form $\mathrm{CT}_\theta = (C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ such that $\mathsf{H}_2(C_{\theta,0}, C_{\theta,1}, C^*_{\theta,2}) = t^*_\theta$, but $\mathrm{CT}_\theta \neq \mathrm{CT}^*_\theta = (C^*_{\theta,0}, C^*_{\theta,1}, C^*_{\theta,2}, C^*_{\theta,3}, C^*_{\theta,4})$ at Step 4. We know that this event generates a collision of the hash function H_2. Hence, we have

$$\Pr[\mathsf{Event}_2] \le \varepsilon_{\mathsf{H}_2,\mathsf{CR}}$$

where $\varepsilon_{\mathsf{H}_2,\mathsf{CR}}$ is the success probability that $\mathcal{A}$ breaks the collision resistance of H_2.

Denote by $\mathsf{Abort}_\mathcal{B}$ the union of the events Event_1, Event_2, and Event_3. Then, we have

$$\Pr[\mathsf{Abort}_\mathcal{B}] = \Pr[\mathsf{Event}_1 \vee \mathsf{Event}_2 \vee \mathsf{Event}_3] \le \Pr[\mathsf{Event}_1 \vee \mathsf{Event}_3] + \Pr[\mathsf{Event}_2] \le q/p + \varepsilon_{\mathsf{H}_2,\mathsf{CR}}.$$

Denote by $\mathsf{Succ}_\mathcal{B}$ the event that $\mathcal{B}$ outputs the correct answer in the DBDH game. Recall that DBDH challenge tuple is (g, g^a, g^b, g^c, T) where $T = T_1 = e(g,g)^{abc}$ or $T = T_0 \xleftarrow{\$} \mathbb{G}_T$. Then, we have

$$\begin{aligned}
\epsilon_{\mathcal{B},\mathsf{DBDH}} &= \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B}] \\
&\ge \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma] \\
&= \frac{1}{4}\Big(\Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_1] + \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 1 \wedge T = T_1] \\
&\quad + \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_0] + \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 1 \wedge T = T_0] \Big) \\
&\ge \frac{1}{4}\Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_1] + \frac{1}{4}\Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 1 \wedge T = T_1] \\
&\ge \frac{1}{2} \cdot \Pr[\mathsf{Succ}_\mathcal{B} \wedge \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_1] \\
&= \frac{1}{2} \cdot \Pr[\mathsf{Succ}_\mathcal{B} \mid \neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \wedge \delta = 0 \wedge T = T_1] \cdot \Pr[\neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_1] \\
&= \frac{1}{2} \cdot 1 \cdot \Pr[\neg\mathsf{Abort}_\mathcal{B} \wedge \gamma' = \gamma \mid \delta = 0 \wedge T = T_1]
\end{aligned}$$

Table 1
Feature comparison of our construction with existing PKEET schemes.

	Security		Model	Assumptions
	OW-CCA2	IND-CCA2		
[1]	✓	✗	ROM	CDH
[4]	✓	✓	ROM	CDH
[6]	✓	✓	ROM	CDH
[10]	✓	✓	SM	Generic
Ours	✓	✓	SM	DBDH

Legend: ROM: random oracle model, SM: standard model, CDH: computational Diffie–Hellman assumption, DBDH: decisional bilinear Diffie–Hellman assumption.

$$
\begin{aligned}
&= \frac{1}{2} \cdot \Pr[\gamma' = \gamma \mid \neg \mathsf{Abort}_{\mathcal{B}} \wedge \delta = 0 \wedge T = T_1] \cdot \Pr[\neg \mathsf{Abort}_{\mathcal{B}} \mid \delta = 0 \wedge T = T_1] \\
&\geq \frac{1}{2} \cdot \mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND-CCA}} \cdot (1 - \Pr[\mathsf{Abort}_{\mathcal{B}} \mid \delta = 0 \wedge T = T_1]) \\
&\geq \frac{1}{2} \cdot \varepsilon_{\mathrm{OURS}} \cdot (1 - 4 \Pr[\mathsf{Abort}_{\mathcal{B}} \wedge \delta = 0 \wedge T = T_1]) \\
&\geq \frac{1}{2} \cdot \varepsilon_{\mathrm{OURS}} \cdot (1 - 4 \Pr[\mathsf{Abort}_{\mathcal{B}}]) \\
&\geq \frac{1}{2} \cdot (1 - 4(\epsilon_{\mathsf{H}_2,\mathsf{CR}} - q/p)) \cdot \varepsilon_{\mathrm{OURS}}.
\end{aligned}
$$

The third and tenth steps hold since $\Pr[\delta = 0 \wedge T = T_0] = \Pr[\delta = 0 \wedge T = T_1] = \Pr[\delta = 1 \wedge T = T_0] = \Pr[\delta = 1 \wedge T = T_1] = 1/4$ where δ is independent of T. In the fourth step, we assume that the former probability is less than the latter one without loss of generality, so the fifth step holds. The seventh step holds because when $\mathcal{B}$ does not abort and $T = e(g, g)^{abc}$ along with $\mathcal{A}$ outputting $\gamma' = \gamma$ and $\delta = 0$, the probability that $\mathcal{B}$ outputs the correct answer is 1. The eighth step holds because when $\mathcal{B}$ does not abort, $T = e(g, g)^{abc}$ and $\delta = 0$, $\mathcal{A}$'s view is identical to its view in the real security game, and thus the probability that $\mathcal{A}$ outputs $\gamma' = \gamma$ is the same as the advantage of $\mathcal{A}$.

Therefore, we obtain

$$\varepsilon_{\mathrm{OURS}} \leq \frac{2}{1 - 4(\varepsilon_{\mathsf{H}_2,\mathsf{CR}} - q/p)} \cdot \varepsilon_{\mathcal{B},\mathsf{DBDH}}. \quad \square$$

5. Comparisons and discussions

In this section, we provide comparisons of our proposed scheme with existing PKEET constructions under the same PKEET system model as ours. In Table 1, we present a feature comparison among them; the third, fourth, fifth, sixth and last rows show features of Yang et al.'s first PKEET scheme [1], Tang's construction [4], Ma et al.'s scheme [6], Lee et al.'s generic construction [10], and our scheme, respectively. Table 1 tells us that our scheme is the first scheme whose security relies on a specific cryptographic assumption in the standard model. We remark that Lee et al.'s work [10] is generic in the sense that it requires generic cryptographic assumptions such as the existence of 2-level HIBE schemes which are secure under selective identity and chosen plaintext attacks, and strongly unforgeable one-time signature schemes. All other listed work achieve security requirements in the random oracle model; we note that Yang et al.'s construction [1] achieves OW-CCA2 security only since anyone can perform equality tests publicly without trapdoors in their scheme.

Table 2 also gives a comparison of space and time complexities among existing PKEET constructions. We employed Boneh and Boyen's 2-level HIBE scheme [11] and Boneh, Shen, and Waters' signature scheme [12] for Lee et al.'s generic construction. We set the output size of exploited hash functions to 2λ bits for security parameter λ. We remark that Ma et al.'s work [6] additionally supports flexible authorization that can issue trapdoors of different authorization levels, e.g., trapdoors for one specific ciphertext or all ciphertexts. Thus, it seems natural that their scheme is somewhat inefficient compared to other listed PKEET schemes in the random oracle model.

We observe that the schemes in the random oracle model are generally more efficient than those in the standard model. However, we confirm that parameter sizes of our construction and performance of our encryption algorithm are comparable to those of the existing efficient schemes [4,6] in the random oracle model. Table 2 also shows that our scheme has better performance than the outcome obtained by Lee et al.'s generic construction: Our work reduces the number of group elements in a ciphertext from linear in λ to constant for security parameter λ. Furthermore, we also reduce computational costs for encryption, decryption, and test algorithms by about 60%, 77%, and 66%, respectively.

Practical consideration for encrypting binary string messages. The message space of our proposed scheme in Section 3 is $\mathbb{G}_T$, which is the image of the employed bilinear map. Generally, it is a subgroup of a multiplicative group of a finite extension field if we exploit a bilinear map defined using elliptic curves. Hence we need an encoding function from the set of binary strings into $\mathbb{G}_T$ to encrypt a binary string message using our scheme. However, to the best of our knowledge, there

Table 2
Efficiency comparison of our construction with existing PKEET schemes.

	Parameter sizes			
	Public key	Ciphertext		Trapdoor
[1]	$\|\mathbb{G}\|$	$3\|\mathbb{G}\| + \|\mathbb{Z}_p\|$		–
[4]	$2\|\mathbb{G}\|$	$4\|\mathbb{G}\| + \|\mathbb{Z}_p\| + 2\lambda$		$\|\mathbb{Z}_p\|$
[6]	$3\|\mathbb{G}\|$	$5\|\mathbb{G}\| + \|\mathbb{Z}_p\|$		$\|\mathbb{Z}_p\|$
[10]	$5\|\mathbb{G}\|$	$(2\lambda + 13)\|\mathbb{G}\| + 2\|\mathbb{G}_T\| + \|\mathbb{Z}_p\|$		$2\|\mathbb{G}\|$
Ours	$3\|\mathbb{G}\| + 2\|\mathbb{G}_T\|$	$2\|\mathbb{G}\| + 2\|\mathbb{G}_T\| + \|\mathbb{Z}_p\|$		$\|\mathbb{G}\|$
	Computational costs			Model
	Encryption	Decryption	Test	
[1]	3Exp	3Exp	2BP	ROM
[4]	5Exp	2Exp	4Exp	ROM
[6]	6Exp	5Exp	2BP + 2Exp	ROM
[10]	1BP + 15Exp	9BP + 11Exp	6BP + 6Exp	SM
Ours	6Exp	2BP + 1Exp	2BP	SM

Legend: $|\mathbb{G}|$, $|\mathbb{G}_T|$, $|\mathbb{Z}_p|$: bit-lengths to represent elements in groups $\mathbb{G}$, $\mathbb{G}_T$, and $\mathbb{Z}_p$, respectively, BP: computational cost for one bilinear map evaluation, Exp: computational cost for an exponentiation operation, ROM: random oracle model, SM: standard model, λ: security parameter.

is no efficient encoding algorithm from a set of binary strings to $\mathbb{G}_T$ if the size of the set of binary strings is exponential in the security parameter. (Recall that the size of the message space of our scheme should be exponential in the security parameter.)

One way to overcome this obstacle is to apply a hybrid encryption technique proposed by Cramer and Shoup [17]. The hybrid encryption technique employs a PKE scheme and a symmetric key encryption (SKE) scheme together. To encrypt a message, the employed PKE scheme encrypts a randomly chosen secret key of the employed SKE scheme (or a random seed for generating a secret key of the employed SKE scheme). Then, the employed SKE scheme encrypts the message under the secret key (or it generated by the seed) encrypted by the PKE scheme. The ciphertext consists of these two ciphertexts generated by the PKE and SKE schemes, respectively.

More concretely, by applying the hybrid encryption technique to our proposed scheme, our encryption algorithm is modified so that C_0 in a ciphertext is replaced by (C_0, C_0') such that

$$C_0 = R \cdot A^s,\ C_0' = \mathsf{SKE.Enc}(\mathsf{PRF}(R), m),$$

where R is a randomly chosen element from $\mathbb{G}_T$, PRF is a pseudo-random function (PRF), and $\mathsf{SKE.Enc}(sk, m)$ is an encryption algorithm of the exploited SKE scheme that takes a secret key sk and a message m as inputs, and returns a ciphertext. (We note that the pre-image of H_1 in our scheme should be also changed from $\mathbb{G}_T$ to the set of binary strings.) The above modification additionally requires one encryption/decryption of the SKE scheme and one PRF evaluation for encryption/decryption each. The ciphertext size of the SKE scheme is also added to the size of a ciphertext of our scheme. We note that the modification maintains the same security level as our original scheme if the employed SKE scheme is CCA2-secure.

6. Conclusions

In this paper, we proposed an efficient PKEET scheme in the standard model. Our construction achieves OW-CCA2 security against Type-I adversaries who have trapdoors for equality tests and IND-CCA2 security against Type-II adversaries who do not have trapdoors, under the DBDH assumption and the one-wayness and collision-resistance of exploited hash functions. Our proposed scheme improves the efficiency of both parameter sizes and computational costs much: The ciphertext size of our scheme is a small constant number of group elements, while that of the previous scheme is linear in the security parameter. The encryption, decryption, and test algorithms are also improved by about 60%, 77%, and 66%, respectively, than the previous scheme.

As far as we know, there is no specific construction of identity-based encryption with equality test (IBEET) in the standard model, except an outcome obtained by Lee et al.'s generic approach [10]. Thus, it would be an interesting research topic to design an efficient IBEET construction in the standard model.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments. This work was partially done while Kai Zhang visited and Hyung Tae Lee was with Nanyang Technological University, Singapore. Kai Zhang, Jie Chen and Haifeng Qian were supported by the National Natural Science Foundation of China (61571191, 61472142) and the Open Foundation of State Key Laboratory of Integrated Services Networks (ISN17-11). Kai Zhang and Haifeng Qian were also supported by the "Dawn" Program of Shanghai Education Commission (No. 16SG21). Hyung Tae Lee and Huaxiong Wang were supported by Research Grant TL-9014101684-01. Hyung Tae Lee was also supported by "Research Base Construction Fund Support Program" funded by Chonbuk National University in 2018. Huaxiong Wang was also supported by Singapore Ministry of Education under Research Grant MOE2016-T2-2-014(S).

References

[1] G. Yang, C.H. Tan, Q. Huang, D.S. Wong, Probabilistic public key encryption with equality test, in: Topics in Cryptology – CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, Proceedings, San Francisco, CA, USA, March 1–5, 2010, Springer, 2010, pp. 119–131.

[2] Q. Tang, Towards public key encryption scheme supporting equality test with fine-grained authorization, in: Information Security and Privacy – 16th Australasian Conference, Proceedings, ACISP 2011, Melbourne, Australia, July 11–13, 2011, Springer, 2011, pp. 389–406.

[3] Q. Tang, Public key encryption schemes supporting equality test with authorisation of different granularity, Int. J. Appl. Cryptogr. 2 (4) (2012) 304–321.

[4] Q. Tang, Public key encryption supporting plaintext equality test and user-specified authorization, Secur. Commun. Netw. 5 (12) (2012) 1351–1362.

[5] S. Ma, M. Zhang, Q. Huang, B. Yang, Public key encryption with delegated equality test in a multi-user setting, Comput. J. 58 (4) (2015) 986–1002.

[6] S. Ma, Q. Huang, M. Zhang, B. Yang, Efficient public key encryption with equality test supporting flexible authorization, IEEE Trans. Inform. Forensics Secur. 10 (3) (2015) 458–470.

[7] H.T. Lee, S. Ling, J.H. Seo, H. Wang, Semi-generic construction of public key encryption and identity-based encryption with equality test, Inform. Sci. 373 (2016) 419–440.

[8] M. Bellare, P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, in: Proceedings of the 1st ACM Conference on Computer and Communications Security, CCS '93, Fairfax, Virginia, USA, November 3–5, 1993, ACM, 1993, pp. 62–73.

[9] R. Canetti, O. Goldreich, S. Halevi, The random oracle methodology, revisited, J. ACM 51 (4) (2004) 557–594.

[10] H.T. Lee, S. Ling, J.H. Seo, H. Wang, T.-Y. Youn, Public Key Encryption with Equality Test in the Standard Model, Report 2016/1182, 2016, cryptology ePrint archive, http://eprint.iacr.org/2016/1182.

[11] D. Boneh, X. Boyen, Efficient selective-ID secure identity-based encryption without random oracles, in: Advances in Cryptology – EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings, Interlaken, Switzerland, May 2–6, 2004, Springer, 2004, pp. 223–238.

[12] D. Boneh, E. Shen, B. Waters, Strongly unforgeable signatures based on computational Diffie–Hellman, in: Public Key Cryptography – PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, Proceedings, New York, NY, USA, April 24–26, 2006, Springer, 2006, pp. 229–240.

[13] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. Inform. Theory 31 (4) (1985) 469–472.

[14] J. Lai, R.H. Deng, S. Liu, W. Kou, Efficient CCA-secure PKE from identity-based techniques, in: Topics in Cryptology – CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, Proceedings, San Francisco, CA, USA, March 1–5, 2010, Springer, 2010, pp. 132–147.

[15] K. Huang, R. Tso, Y. Chen, S.M.M. Rahman, A. Almogren, A. Alamri, PKE-AET: public key encryption with authorized equality test, Comput. J. 58 (10) (2015) 2686–2697.

[16] H.T. Lee, S. Ling, J.H. Seo, H. Wang, CCA2 attack and modification of Huang et al.'s public key encryption with authorized equality test, Comput. J. 59 (11) (2016) 1689–1694.

[17] R. Cramer, V. Shoup, Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack, SIAM J. Comput. 33 (1) (2003) 167–226.