EPiC Series in Computing

Volume 106, 2025, Pages 122–132

Proceedings of the 20th Asia Joint Conference on Information Security

Tightly Secure Public-Key Encryption with Equality Test Supporting Flexible Authorization in the Standard Model

Yi-Fan Tseng[1]*, Yi-Jiin Lu[1], Tien-Lin Tsai[1], and Zi-Yuan Liu[2]

[1] National Chengchi University, Taipei, Taiwan
[2] National Central University, Taoyuan, Taiwan

Abstract

We introduce a novel Public Key Encryption with Equality Test supporting Flexible Authorization scheme offering User-Level, Ciphertext-Level, and User-Specific-Ciphertext-Level authorizations. Notably, our construction achieves security under the Decisional Diffie-Hellman assumption with a tight reduction, whereas the existing works are either not tightly secure or rely heavily on the random oracles. By relying solely on the standard DDH assumption, our scheme offers practical implementation without specialized cryptographic structures.

1 Introduction

The fundamental importance of encryption [22] in safeguarding sensitive data within increasingly interconnected digital environments cannot be overstated. As data proliferates across cloud platforms, databases, and collaborative networks, the ability to perform meaningful operations on this encrypted information without compromising its confidentiality becomes paramount. Public Key Encryption with Equality Test (PKEET), which was first conceptualized in [25], emerges as a crucial cryptographic primitive in addressing this challenge. Specifically, PKEET enables the verification of whether two independently encrypted datasets contain the same underlying plaintext, a capability essential for applications like efficient cloud data deduplication [12, 21, 27] to minimize storage overhead, privacy-preserving database [26, 29] querying to identify matching encrypted records, etc. Lots of PKEET [5, 10, 11, 13, 16] have been proposed in literature since the work of [25]. However, the varied security and operational requirements of these diverse applications necessitate a more flexible approach to controlling who can perform these equality tests and under what circumstances, thereby highlighting the critical need for adaptable authorization mechanisms within PKEET schemes. Therefore, the concept of PKEET supporting flexible authorization (PKEET-FA, for short) has been proposed in [17].

In the context of cloud data deduplication, diverse needs necessitate different authorization strategies for equality tests on encrypted data. For scenarios where a service provider aims for efficient deduplication within a trusted group, allowing any encrypted file of a user within that group to be compared against all encrypted files of other users in the same group proves beneficial. This approach streamlines storage and collaboration among team members. In [17], this case is named **User-Level Authorization**.

*Corresponding Author: yftseng@g.nccu.edu.tw

Conversely, users might desire finer control, wanting only specific encrypted files to be considered for deduplication. By associating a unique identifier with these particular encrypted files, equality tests are restricted to files sharing the same identifier. This granular control over deduplication participation is **Ciphertext-Level Authorization** in [17]. In highly specific collaborative settings, users might need to limit comparisons even further, allowing a particular encrypted file of theirs to be tested only against a designated specific encrypted file belonging to another specific user, catering to very targeted data sharing. This highly restricted comparison is **User-Specific-Ciphertext-Level Authorization** in [17]. Lastly, in scenarios involving potentially public or community-driven content, a user might want their specific encrypted file to be compared against all encrypted files uploaded by any other user on the platform to identify potential duplicates across a wider user base. This broader comparison initiated from a single ciphertext is **Ciphertext-to-User-Level Authorization** in [17].

1.1 Related Works

The first Public Key Encryption with Equality Test (PKEET) scheme was introduced by Ma et al. in 2014 [17]. As previously discussed, their work also proposed four distinct authorization models for PKEET. The scheme presented in [17] is constructed over pairing groups and its security is proven within the random oracle model. To address the performance limitations often associated with pairing-based constructions, Lin et al. proposed a pairing-free PKEET scheme in 2021 [15]. However, it's worth noting that their scheme's security is also proven under the random oracle model. On the other hand, Nguyen *et al.* [20] and Li [14] give the ID-based variant of PKEET-FA, named IBEET-FA. While the ID-based setting elegantly eliminates the certificate authority and its associated management overhead, it inherently suffers from the key-escrow problem, where a central Private Key Generator (PKG) possesses the ability to generate the private keys of all users. This fundamental issue restricts the applicability of IBEET-FA primarily to smaller-scale deployments due to the potential for a single point of failure. Furthermore, the schemes presented in [20] and [14] exhibit additional drawbacks compared to other approaches. The construction in Nguyen et al. [20], while offering resistance to quantum attacks due to its foundation in lattice cryptography, typically incurs a performance penalty due to the large parameter sizes inherent in lattice-based systems. As for Li's work [14], it relies on a specialized algebraic structure known as Trapdoor Discrete-Log (TDL) Groups, where the group generator holds a trapdoor enabling the efficient solution of the discrete logarithm problem within the group. Unfortunately, TDL groups currently lack practical and widely adopted implementations. A significant challenge in existing PKEET-FA schemes lies in the security proofs, which are often established either under the random oracle model or are non-tight. While the random oracle model is widely regarded as a useful heuristic within the realm of provable security, it is crucial to acknowledge that it lacks a concrete realization in practical systems. Furthermore, notable counterexamples [4, 18] have demonstrated that cryptographic primitives proven secure in the random oracle model may not retain their security properties in real-world deployments. Consequently, the standard model, which imposes no additional restrictions on the adversary's computational capabilities, is a more desirable framework for security proofs. Regarding tightness, a tight security reduction ensures that the security level of the scheme is directly linked to the hardness of its underlying computational assumption. This allows for the selection of smaller security parameters, leading to improved performance compared to schemes with non-tight security proofs.

1.2 Contribution

After our survey, existing PKEET-FA schemes exhibit certain limitations, which serves as the primary motivation for our work. In this manuscript, we propose a novel PKEET-FA scheme that offers the following advantages:

- **Flexible Authorization Support**: Our scheme natively supports three distinct authorization models: User-Level Authorization, Ciphertext-Level Authorization, and User-Specific-Ciphertext-Level Authorization. This comprehensive support allows for a wider range of application scenarios compared to schemes supporting fewer authorization types.
- **Provable Security in the Standard Model with Tight Reduction**: The security of our proposed scheme is rigorously proven under the Decisional Diffie-Hellman (DDH) assumption within the standard model, eliminating the reliance on the heuristic random oracle model. Furthermore, to achieve tight security, our security proof employs a technique inspired by [16], which strategically embeds the instance of the DDH assumption into the public keys of all users, rather than a specific one. This tight security reduction provides stronger security guarantees and potentially allows for the use of smaller security parameters without compromising the security level.

2 Preliminaries

2.1 Definition for PKEET-FA

In this section, we give the definition of a PKEET scheme with flexible authorization. A PKEET scheme consists of a tuple of (at least) six algorithms $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Aut}, \mathsf{Test})$. $\mathsf{Setup}(1^\lambda)$ algorithm takes as input a security parameter $\lambda \in \mathbb{N}$, and outputs a public parameter pp, which will be an implicit input to all the following algorithms. The algorithm $\mathsf{KG}(\mathsf{pp})$ is performed by each user to generate her/his own public/private key pair $(\mathsf{PK}, \mathsf{SK})$. $\mathsf{Enc}(\mathsf{PK}, \mathsf{M})$ takes inputs a public key PK and a message M, and outputs a ciphertext CT. $\mathsf{Dec}(\mathsf{PK}, \mathsf{SK}, \mathsf{M})$ then takes as inputs $(\mathsf{PK}, \mathsf{SK})$ and a ciphertext to output a message M or an error symbol $\perp$. As for the authorization algorithm Aut and the test algorithm Test, we adopt the definition of flexible authorization given in [17]. Let U_i be a user indexing by a positive integer i. There are four types of authorizations, each type has their own Aut and Test algorithms:

- **User-Level Authorization**: All the ciphertext of a specific user can be compared to all the ciphertexts of any other users. In this type of authorization,
  - $\mathsf{Aut}_1(\mathsf{SK}_i)$ generates a token $\mathsf{tok}_i^{(1)}$ upon inputting the secret key of a user U_i;
  - $\mathsf{Test}_1(\mathsf{CT}_i, \mathsf{tok}_i^{(1)}, \mathsf{CT}_j, \mathsf{tok}_j^{(1)})$ takes U_i's ciphertex CT_i and her token $\mathsf{tok}_i^{(1)}$, and U_j's ciphertext CT_j and her token $\mathsf{tok}_j^{(1)}$, outputs 1 if CT_i and CT_j is the encryption of the same message; output 0 otherwise.
- **Ciphertext-Level Authorization**: In this type of authorization, A token is bound with a specific ciphertext, and the test algorithm is then used to check whether two specific ciphertexts encrypts the same message. Formally,
  - $\mathsf{Aut}_2(\mathsf{SK}_i, \mathsf{CT}_i)$ takes U_i's private key SK_i and a U_i's ciphertext CT_i, outputs $\mathsf{tok}_{\mathsf{CT}_i}^{(2)}$ relates to CT_i;
  - $\mathsf{Test}_2(\mathsf{CT}_i, \mathsf{tok}_{\mathsf{CT}_i}^{(2)}, \mathsf{CT}_j, \mathsf{tok}_{\mathsf{CT}_j}^{(2)})$ outputs 1 if CT_i and CT_j is the encryption of the same message, and output 0 otherwise.
- **User-Specific-Ciphertext-Level Authorization**: A specific ciphertext of a user U_i can only be compared with a specific ciphertext of a specific user U_j. In this case,
  - $\mathsf{Aut}_3(\mathsf{SK}_i, \mathsf{CT}_i, \mathsf{PK}_j, \mathsf{CT}_j)$ takes as inputs U_i's private key SK_i and a ciphertext CT_i, and U_j's public key PK_i and a ciphertext CT_j, outputs a token $\mathsf{tok}_{\mathsf{CT}_i,\mathsf{CT}_j}^{(3)}$;

  - $\mathsf{Test}_3(\mathsf{CT}_i, \mathsf{tok}^{(3)}_{\mathsf{CT}_i,\mathsf{CT}_j}, \mathsf{CT}_j, \mathsf{tok}^{(3)}_{\mathsf{CT}_j,\mathsf{CT}_i})$ outputs 1 if CT_i and CT_j is the encryption of the same message, and output 0 otherwise.

- **Ciphertext-to-User-Level Authorization**: A specific ciphertext CT_i of U_i can be compared with all the ciphertext of any other user. As claimed in [17], such setting can be achieved by simply combining **User-Level Authorization** and **Ciphertext-Level Authorization**. For instance, If U_i wishes to authorize the cloud to check whether a ciphertext CT_i encrypts a message identical to any ciphertext of user U_j, then we require U_i runs $\mathsf{tok}^{(2)}_{\mathsf{CT}_i} \leftarrow \mathsf{Aut}_2(\mathsf{SK}_i, \mathsf{CT}_i)$, and U_j runs $\mathsf{tok}^{(1)}_j \leftarrow \mathsf{Aut}_1(\mathsf{SK}_j)$. Then Test_4 is the combination of Test_1 and Test_2.

2.2 Security Models for PKEET-FA

In PKEET-FA, two types of adversaries are considered:

- Type-I Adversary $\mathcal{A}_I$: This type of adversary is allowed to obtained tokens for equality test, and its goal is to recover the encrypted message from a given challenge ciphertext.
- Type-II Adversary $\mathcal{A}_{II}$: This type of adversary is an analogue to the CPA/CCA adversary of public-key encryption, where an adversary given a challenge ciphertext, which is the encryption of one of the two massage chosen by the adversary, is asked to determine which of the two messages is encrypted. A natural restriction then rises from the functionality of PKEET, that is $\mathcal{A}_{II}$ is not allow query the token of the challenge ciphertext.

In the following we show two security games defining the security against Type-I and Type-II adversaries respectively. In both the security games there are two roles in this game, a Type-I adversary $\mathcal{A}_I$ and a challenger $\mathcal{C}$. In order to reduce the literal repetition, we first describe several oracle that the adversary would query in the security games. A list L_{user} is maintained by $\mathcal{C}$ to record the public/private key of users.

- $\mathcal{O}_{\mathsf{create}}(i)$: Upon inputting an index $i \in \mathbb{N}$, $\mathcal{C}$ runs $(\mathsf{PK}_i, \mathsf{SK}_i) \leftarrow \mathsf{KG}(\mathsf{pp})$ and record $(\mathsf{PK}_i, \mathsf{SK}_i, \mathsf{tag}_i = \mathsf{honest})$ in L_{user}.
- $\mathcal{O}_{\mathsf{corr}}(i)$: Upon inputting an index i, the oracle returns SK_i and updates the record $(\mathsf{PK}_i, \mathsf{SK}_i, \mathsf{tag}_i = \mathsf{corrupt})$ in L_{user}.
- $\mathcal{O}_{\mathsf{Aut}}$: The input of this oracle contains an index $i = 1, 2, 3$, which indicates the authorization types, and a string inp_i, which depends on the authorization type. Formally, for $i = 1, 2, 3$,
  - $\mathsf{inp}_1 = i$ and the oracle returns $\mathsf{tok}^{(1)}_i \leftarrow \mathsf{Aut}_1(\mathsf{SK}_i)$;
  - $\mathsf{inp}_2 = (i, \mathsf{CT}_i)$ and the oracle returns $\mathsf{tok}^{(2)}_{\mathsf{CT}_i} \leftarrow \mathsf{Aut}_2(\mathsf{SK}_i, \mathsf{CT}_i)$;
  - $\mathsf{inp}_3 = (i, \mathsf{CT}_i, j, \mathsf{CT}_j)$ and the oracle returns $\mathsf{tok}^{(3)}_{\mathsf{CT}_i,\mathsf{CT}_j} \leftarrow \mathsf{Aut}_3(\mathsf{SK}_i, \mathsf{CT}_i, \mathsf{PK}_j, \mathsf{CT}_j)$.
- $\mathcal{O}_{\mathsf{Dec}}(i, \mathsf{CT}_i)$: Upon inputting an index i and a ciphertext CT_i, the oracle outputs the result of $\mathsf{Dec}(\mathsf{SK}_i, \mathsf{CT}_i)$.

We then give the following two security games.

OW-CCA Game: This game consists of the following phases.

1. Setup: The challenger $\mathcal{C}$ runs $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and sends pp to the adversary $\mathcal{A}_I$. A list L_{user} is maintained by $\mathcal{C}$.
2. Phase 1: In this phase, the adversary is allowed to query the oracles $\mathcal{O}_{\mathsf{create}}, \mathcal{O}_{\mathsf{corr}}, \mathcal{O}_{\mathsf{Aut}}, \mathcal{O}_{\mathsf{Dec}}$.
3. Challenge: $\mathcal{A}_I$ submits an index i^* with restriction that $\mathsf{tag}_{i^*} = \mathsf{honest}$ in L_{user}. $\mathcal{C}$ then chooses a random message M^* and returns $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{PK}_{i^*}, \mathsf{M}^*)$.
4. Phase 2: The adversary is allowed to query the oracles identical to those in Phase 1, except for $\mathcal{O}_{\mathsf{corr}}(i^*)$ and $\mathcal{O}_{\mathsf{Dec}}(i^*, \mathsf{CT}^*)$.
5. Guess: $\mathcal{A}_I$ outputs a message M', and wins the game if $\mathsf{M}' = \mathsf{M}^*$.

The advantage for $\mathcal{A}_I$ in winning the game is defined as

$$\mathsf{Adv}^{\text{OW-CCA}}_{\mathcal{A}_I}(\lambda) = \Pr[\mathsf{M}' = \mathsf{M}^*].$$

Definition 1 (The One-wayness against Chosen-Ciphertext Attacks)**.** *We say that a PKEET-FA scheme satisfies the one-wayness against chosen-ciphertext attacks (OW-CCA) if for all PPT Type-I adversary $\mathcal{A}_I$, $\mathsf{Adv}^{\text{OW-CCA}}_{\mathcal{A}_I}(\lambda)$ is negligible.*

IND-CCA Game: This game consists of the following phases.

1. Setup: The challenger $\mathcal{C}$ runs $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and sends pp to the adversary $\mathcal{A}_{II}$. A list L_{user} is maintained by $\mathcal{C}$.
2. Phase 1: In this phase, the adversary is allowed to query the oracles $\mathcal{O}_{\mathsf{create}}, \mathcal{O}_{\mathsf{corr}}, \mathcal{O}_{\mathsf{Aut}}, \mathcal{O}_{\mathsf{Dec}}$.
3. Challenge: $\mathcal{A}_{II}$ submits an index i^* and two distinct messages $\mathsf{M}_0, \mathsf{M}_1$, with restriction that $\mathcal{O}_{\mathsf{corr}}(i^*)$ and $\mathcal{O}_{\mathsf{Aut}}(1, (i^*))$ have not been queried. $\mathcal{C}$ then selects $b \xleftarrow{\$} \{0, 1\}$ and returns $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{PK}_{i^*}, \mathsf{M}_b)$.
4. Phase 2: The adversary is allowed to query the oracles identical to those in Phase 1, except for $\mathcal{O}_{\mathsf{corr}}(i^*)$, $\mathcal{O}_{\mathsf{Dec}}(i^*, \mathsf{CT}^*)$, $\mathcal{O}_{\mathsf{Aut}}(1, (i^*))$, $\mathcal{O}_{\mathsf{Aut}}(2, (i^*, \mathsf{CT}^*))$, and $\mathcal{O}_{\mathsf{Aut}}(3, (i^*, \mathsf{CT}^*, \cdot, \cdot))$.
5. Guess: $\mathcal{A}_{II}$ outputs a bit b', and wins the game if $b' = b$.

The advantage for $\mathcal{A}_{II}$ in winning the game is defined as

$$\mathsf{Adv}^{\text{IND-CCA}}_{\mathcal{A}_{II}}(\lambda) = \Pr[b' = b] - \frac{1}{2}.$$

Definition 2 (The Indistinguishability against Chosen-Ciphertext Attacks)**.** *We say that a PKEET-FA scheme satisfies the indistinguishability against chosen-ciphertext attacks (IND-CCA) if for all PPT Type-II adversary $\mathcal{A}_{II}$, $\mathsf{Adv}^{\text{IND-CCA}}_{\mathcal{A}_{II}}(\lambda)$ is negligible.*

3 Our Construction

In this section, we show the proposed PKEET-FA scheme as follows. Our scheme is constructed based on the PKEET scheme shown in [16], which achieves only CPA security. As we claimed in the previous papers, ciphertext-to-user-level authorization can be obtained from others, and hence we omit it here due to the length limitation.

$\mathsf{Setup}(1^\lambda)$.

1. Generate the description for a multiplicative group $(\mathbb{G}, q, g)$, where $|\mathbb{G}| = q$ and g generates $\mathbb{G}$.
2. Choose $\alpha \xleftarrow{\$} \mathbb{Z}_q$ and compute $h = g^\alpha$.
3. Select a cryptographic hash function $H : \{0,1\}^* \to \mathbb{G}$.
4. Choose a strongly unforgeable OTS $\Sigma = (\Sigma.\mathsf{KG}, \Sigma.\mathsf{Sign}, \Sigma.\mathsf{Vrf})$
5. Output $\mathsf{pp} = (\mathbb{G}, q, g, h, H, \Sigma)$.

$\mathsf{KG}(\mathsf{pp})$.

1. Choose $x, y, u, v \xleftarrow{\$} \mathbb{Z}_q$.
2. Compute $K = g^x h^y = g^{x+\alpha y}, K' = g^u h^v = g^{u+\alpha v}$.
3. Output $\mathsf{PK} = (K, K')$ and $\mathsf{SK} = (x, y, u, v)$.

$\mathsf{Enc}(\mathsf{PK}, \mathsf{M})$.

1. Run $(\mathsf{vk}, \mathsf{sk}) \leftarrow \Sigma.\mathsf{KG}(1^\lambda)$.
2. Choose randomly $s \xleftarrow{\$} \mathbb{Z}_q$.
3. Compute $E_1 = g^s, E_2 = h^s, E_3 = K^s \cdot (\mathsf{M} \| \mathsf{vk}), E_4 = (K')^s \cdot (H(\mathsf{M}) \| \mathsf{vk}), E_5 = \Sigma.\mathsf{Sign}(\mathsf{sk}, (E_1, \ldots, E_4))$.
4. Output $\mathsf{CT} = (E_1, E_2, E_3, E_4, E_5)$.

$\mathsf{Dec}(\mathsf{SK}, \mathsf{CT})$.

1. Compute $(\mathsf{M}' \| R') = E_3/(E_1^x E_2^y)$,
   $(h' \| R'') = E_4/(E_1^u E_2^v)$.
2. Output M' if $(h' = H(\mathsf{M}')) \wedge (R' = R'') \wedge (\Sigma.\mathsf{Vrf}(\mathsf{vk}, E_5, (E_1, \ldots, E_4)) = 1)$; output $\perp$ otherwise.

User-Level Authorization:
$\mathsf{Aut}_1(\mathsf{SK}_i)$. Parse $\mathsf{SK}_i = (x_i, y_i, u_i, v_i)$ and output $\mathsf{tok}_i^{(1)} = (u_i, v_i)$.

$\mathsf{Test}_1(\mathsf{CT}_i, \mathsf{tok}_i^{(1)}, \mathsf{CT}_j, \mathsf{tok}_j^{(1)})$.

1. Parse $\mathsf{tok}_i^{(1)} = (u_i, v_i), \mathsf{tok}_j^{(1)} = (u_j, v_j)$.
2. Parse $\mathsf{CT}_i = (E_{i,1}, E_{i,2}, E_{i,3}, E_{i,4}, E_{i,5})$, $\mathsf{CT}_j = (E_{j,1}, E_{j,2}, E_{j,3}, E_{j,4}, E_{j,5})$.
3. Compute $(h_i \| R_i) = E_{i,4}/(E_{i,1}^{u_i} E_{i,2}^{v_i}), (h_j \| R_j) = E_{j,4}/(E_{j,1}^{u_j} E_{j,2}^{v_j})$.
4. Output 1 if $h_i = h_j$; output 0 otherwise.

Ciphertext-Level Authorization:
$\mathsf{Aut}_2(\mathsf{SK}_i, \mathsf{CT}_i)$. Parse $\mathsf{CT}_i = (E_{i,1}, E_{i,2}, E_{i,3}, E_{i,4}, E_{i,5}), \mathsf{SK}_i = (x_i, y_i, u_i, v_i)$, and output $\mathsf{tok}_{\mathsf{CT}_i}^{(2)} = E_{i,1}^{u_i} E_{i,2}^{v_i}$.
$\mathsf{Test}_2(\mathsf{CT}_i, \mathsf{tok}_{\mathsf{CT}_i}^{(2)}, \mathsf{CT}_j, \mathsf{tok}_{\mathsf{CT}_j}^{(2)})$.

1. Parse $\mathsf{CT}_i = (E_{i,1}, E_{i,2}, E_{i,3}, E_{i,4}, E_{i,5})$, $\mathsf{CT}_j = (E_{j,1}, E_{j,2}, E_{j,3}, E_{j,4}, E_{j,5})$.
2. Compute $(h_i \| R_i) = E_{i,4}/\mathsf{tok}_{\mathsf{CT}_i}^{(2)}, (h_j \| R_j) = E_{j,4}/\mathsf{tok}_{\mathsf{CT}_j}^{(2)}$.
3. Output 1 if $h_i = h_j$; output 0 otherwise.

4 Security Analysis

Due to the page limitation, we only give the intuition of our security proof in this section.

Theorem 1. *The proposed scheme is OW-CCA secure against Type-I adversary if the DDH assumption holds for $\mathbb{G}$, H is one-way, and Σ is strongly unforgeable.*

Proof. This theorem can be proven via a sequence of games as follows.
Let $\mathsf{CT}^* = (E_1^*, E_2^*, E_3^*, E_4^*, E_5^*)$ be the challenge ciphertext send to the adversary $\mathcal{A}_I$.

- Game 0: This game is identical to the OW-CCA game. That is, the challenger $\mathcal{C}$ generates all the parameters following the algorithms shown in Section 3. Let $\mathsf{CT}^* = (E_1^*, E_2^*, E_3^*, E_4^*, E_5^*)$ where $E_1^* = g^s, E_2^* = h^s, E_3^* = K^s \cdot (\mathsf{M}^* \| \mathsf{vk}^*)$, $E_4 = (K')^s \cdot (H(\mathsf{M}^*) \| \mathsf{vk}^*), E_5^* = \Sigma.\mathsf{Sign}(\mathsf{sk}^*, (E_1^*, \ldots, E_4^*))$.
- Game 1: This game is identical to Game 0, except for the way to answer $\mathcal{O}_{\mathsf{Dec}}$ query. Now the challenger answers the query as follows.
  1. Parse $\mathsf{CT} = (E_1, E_2, E_3, E_4, E_5)$.
  2. Search $\mathsf{PK}_{i^*} = (x_{i^*}, y_{i^*}, u_{i^*}, v_{i^*}, z_{i^*})$ from L_{user}.
  3. Recover $(\mathsf{M}' \| R')$ and $(h' \| R'')$ as the same way to Dec algorithm shown in Section 3.
  4. Output $\perp$ if $R' \neq R''$ or $h' \neq H(\mathsf{M}')$.
  5. Output $\perp$ if $(E_1, E_2, E_3, E_4) \neq (E_1^*, E_2^*, E_3^*, E_4^*)$, $R' = \mathsf{vk}^*$, and $\Sigma.\mathsf{Vrf}(R', E_5, (E_1, E_2, E_3, E_4)) = 1$.
  6. Otherwise, output M'.
- Game 2: This game is identical to Game 1, except that, $E_2^* \leftarrow g^{\tilde{s}}, E_3^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (\mathsf{M}^* \| \mathsf{vk}^*), E_4^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (H_1(\mathsf{M}^*) \| \mathsf{vk}^*)$, for some $\tilde{s} \xleftarrow{\$} \mathbb{Z}_q$.
- Game 3: This game is identical to Game 2, except that, $E_3^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (\tilde{\mathsf{M}} \| \mathsf{vk}^*)$ for a uniformly chosen message $\tilde{\mathsf{M}}$.

It is not hard to see that, in Game 3, the messages used in E_3^* and E_4^* are independent of each others, and hence we can reduce the one-wayness of H to breaking Game 3. That is, an adversary wins in Game 3 can be transformed into an algorithm breaking the one-wayness of H. What remains is to show the indistinguishability between each pair of adjacent games. Let Ev_i be the event that $\mathcal{A}_I$ wins in Game i for $i = 0, 1, 2, 3$. Observing that, in Game 1 the decryption oracle outputs $\perp$ when there are two one-time signatures with respect to the same verification key vk^*, which obviously violates the strong unforgrability of Σ. Thus, according to the Difference lemma shown in [24], we have $|\Pr[\mathsf{Ev}_0] - \Pr[\mathsf{Ev}_1]| \leq \mathsf{Adv}^{\mathrm{SUF}}_{\mathcal{C}_{01}}(\lambda)$. Besides, as $(g, E_1^* = g^s, h = g^\alpha, E_2^* = h^s)$ si a DDH tuple, Game 1 and 2 are indistinguishable due to the DDH assumption in $\mathbb{G}$. As for the indistinguishability between Game 2 and 3, we give the following lemma.

Lemma 1. $|\Pr[\mathsf{Ev}_2] - \Pr[\mathsf{Ev}_3]| \leq \frac{1}{q}$

Proof. This lemma can be proven by showing that E_3^* in Game 2 is exactly a one-time pad. Given CT^* and PK_{i^*}, one can observe that the exponent of the ciphertext component E_3^* and public key component K_{i^*} satisfy the following equation

$$\begin{pmatrix} x_{i^*} + y_{i^*} \cdot \alpha \\ x_{i^*} \cdot s + y_{i^*} \cdot \tilde{s} \end{pmatrix} = \begin{pmatrix} 1 & \alpha \\ s & \tilde{s} \end{pmatrix} \begin{pmatrix} x_{i^*} \\ y_{i^*} \end{pmatrix}.$$

Since x_{i^*}, y_{i^*} are uniformly chosen at random from $\mathbb{Z}_q$, we have that $(x_{i^*} + y_{i^*} \cdot \alpha, x_{i^*} \cdot s + y_{i^*} \cdot \tilde{s})$ is indistinguishable from uniformly random element in $\mathbb{Z}_q^2$, if $\begin{pmatrix} 1 & \alpha \\ s & \tilde{s} \end{pmatrix}$ is invertible. This is equivalent to the statement $\tilde{s} \neq \alpha s \pmod q$. In other word, Game 2 and Game 3 are indistinguishable from $\mathcal{A}_I$'s view except that $\tilde{s} = \alpha s \pmod q$, which holds only with probability $\frac{1}{q}$. □

Finally, we conclude the proof as follows.

$$\mathsf{Adv}^{\text{OW-CCA}}_{\mathcal{A}_I}(\lambda) \leq \mathsf{Adv}^{\text{SUF}}_{\mathcal{C}_{01}}(\lambda) + \mathsf{Adv}^{\text{DDH}}_{\mathcal{C}_{12}}(\lambda) + 1/q + \mathsf{Adv}^{\text{OW}}_{\mathcal{C}_3}(\lambda).$$

□

Theorem 2. *The proposed scheme is IND-CCA secure against Type-II if the DDH assumption holds for* $\mathbb{G}$ *and* Σ *is strongly unforgeable.*

Proof. The proof of Theorem 2 is very similar to that of Theorem 1, which can be proven via the following game sequence.

- Game 0: This game is identical to the IND-CCA game.
- Game 1: This game is identical to Game 0, except the way to answer $\mathcal{O}_{\mathsf{Dec}}$ query. The challenger answers $\mathcal{O}_{\mathsf{Dec}}$ in the same way shown in the proof of Theorem 1. We omit the detail due to the page limitation.
- Game 2: This game is identical to Game 1, except that, $E_2^* \leftarrow g^{\tilde{s}}, E_3^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (\mathsf{M}_b \| \mathsf{vk}^*), E_4^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (H_1(\mathsf{M}_b) \| \mathsf{vk}^*)$, for some $\tilde{s} \xleftarrow{\$} \mathbb{Z}_q$.
- Game 3: This game is identical to Game 2, except that, $E_3^* \leftarrow g^{x_{i^*}s + y_{i^*}\tilde{s}} \cdot (\tilde{\mathsf{M}} \| \mathsf{vk}^*), E_4^* \leftarrow g^{u_{i^*}s + v_{i^*}\tilde{s}} \cdot (H(\tilde{\mathsf{M}}) \| \mathsf{vk}^*)$ for a uniformly chosen message $\tilde{\mathsf{M}}$.

Since in Game 3, CT^* is generated independently of the bit b, $\mathcal{A}_{II}$ only makes a right guess on b with probability $\frac{1}{2}$. The proofs for the indistinguishability between each pair of adjacent games are similar to that shown in the proof of Theorem 1, we directly give the following conclusion and omit the proof due to the page limitation.

$$\mathsf{Adv}^{\text{IND-CCA}}_{\mathcal{A}_{II}}(\lambda) \leq \mathsf{Adv}^{\text{SUF}}_{\mathcal{C}_{01}}(\lambda) + \mathsf{Adv}^{\text{DDH}}_{\mathcal{C}_{12}}(\lambda) + \frac{1}{q}.$$

□

5 Comparison

In this section, we demonstrate the comparison between our work and others supporting flexible authorization [14, 15, 17, 20]. First, we show the performance comparison in Table 1, where shows the comparison in terms of $|\mathsf{CT}|, |\mathsf{SK}|$, and the computation cost of $\mathsf{Enc}, \mathsf{Dec}, \mathsf{Test}_1, \mathsf{Test}_2$ algorithms. We further convert each values in terms of numbers of bytes and clock cycles, in order to give a comprehensible result. For the hash functions in each scheme, we choose SHA3-256 [7], and hence ℓ_H s set to be 32 bytes. Besides, we choose hash-based signature SPHINCS+ [3, 6] as the candidate for OTS. According to [6, 9], $\ell_{\mathsf{OTS}} = 7856$ bytes and $T_{\mathsf{Sign}}, T_{\mathsf{Vrf}}$ are $112937, 13117$ clock cycles for 128-bit security, respectively. For the pairing function, we choose Tate pairing over MNT curves with 80-bit

security. As shown in the results of [2, 8], we have $|\mathbb{G}| = |\mathbb{Z}_q| \approx 37.25$ bytes, and the execution time of a pairing is 13.5 times to that of a scalar in $\mathbb{G}$ under the implementation environment with Core-i7 3.4 GHz processor and 16GB RAM, i.e., $T_P \approx 13.5T_S$. As for the scalar operation, a recent research [28] shows that, such an operation in $\mathsf{GF}(2^k)$ requires $4k + 14$ clock cycles. Besides, according to [19, 23], a modular multiplication for k-bit numbers is about $3k$ clock cycles. Therefore, T_S needs 1206 clock cycles, and T_{mul} needs 894 clock cycles. As for the lattice-based construction, we refer to the parameter set shown in [1], which is a LWE-based KEM submitted to NIST PQC competition. In [1], q and n are set to 4096 bytes and 640 for 128-bit security. Note that, as the large value of q in [20], T_{mul} here needs 98304 clock cycles. Finally, each parameter in [20] depends on $|\mathsf{M}|$, and thus we need to decide the message size of their work. Since the one-wayness of a PKEET is highly related to the size of message space, we set $|\mathsf{M}| = 80$ bits to satisfy 80-bit security. As shown in Table 1, the ciphertext size $|\mathsf{CT}|$ and Enc/Dec performance may not own advantage compared to [14, 15, 17], due to the usage of OTS. Nevertheless, the performance of Test_1 and Test_2 are comparable to others. We then give the property comparison to others in Table 2. First, we should explain some abbreviations in the table. We use "ROM" and "STD" to denote "random oracle model" and "standard model", respectively. By "CDH groups" and "DDH groups" we mean "the groups where the CDH/DDH assumption holds". The notion "TDL group" means a group where the group builder holds a trapdoor to solve the discrete-log problem. All the five scheme support flexible authorization. Since [14, 20] are constructed in ID-based setting, and thus their schemes suffer from the key escrow problem, which may limit the scenario for use due to the single point of failure. Besides, as TDL groups have no practical implementation now, the practical value of [14] is limited. Besides, only ours and [20] are proven secure in the standard model, which is a more desirable result in provable security. In a word, though our scheme may not gain advantage in performance against the others, our work achieves better security guarantee.

	[17]	[20]	[15]	[14]	Ours
$\lvert\mathsf{CT}\rvert$	$2\lvert\mathbb{G}\rvert + 2\ell_H$ ≈ 106.5	$(2\lvert\mathsf{M}\rvert + 6m + m^2)\lvert\mathbb{Z}_q\rvert + \ell_H$ $\approx 1,694,105,632$	$2\lvert\mathbb{G}\rvert + 2\ell_H$ ≈ 106.5	$\lvert\mathbb{G}\rvert + 2\ell_H$ ≈ 69.25	$4\lvert\mathbb{G}\rvert + \ell_{\mathsf{OTS}}$ $\approx 8,005$
$\lvert\mathsf{SK}\rvert$	$3\lvert\mathbb{Z}_q\rvert$ ≈ 111.75	$(8m^2)\lvert\mathbb{Z}_q\rvert$ $\approx 2,097,152,00$	$2\lvert\mathbb{Z}_q\rvert$ ≈ 74.5	$2\lvert\mathbb{Z}_q\rvert$ ≈ 74.5	$4\lvert\mathbb{Z}_q\rvert$ ≈ 149
Enc	$6T_S$ $\approx 4,824$	$(2n\lvert\mathsf{M}\rvert + 3mn + 4m^2)T_{mul}$ $\approx 291,923,558,400$	$4T_S$ $\approx 4,824$	$3T_S$ $\approx 3,618$	$4T_S + T_{\mathsf{Sign}}$ $\approx 117,761$
Dec	$5T_S$ $\approx 6,030$	$(2m\lvert\mathsf{M}\rvert)T_{mul}$ $\approx 10,066,329,600$	$3T_S$ $\approx 3,618$	$2T_S$ $\approx 24,12$	$6T_S + T_{\mathsf{Vrf}}$ $\approx 20,353$
Test_1	$2T_S + 2T_P$ $\approx 34,974$	$(2m\lvert\mathsf{M}\rvert)T_{mul}$ $\approx 10,066,329,600$	$2T_S + 15T_{mul}$ $\approx 15,822$	$2T_S + 2T_{mul}$ $\approx 4,200$	$6T_S$ $\approx 7,236$
Test_2	$2T_P$ $\approx 32,562$	$(2m\lvert\mathsf{M}\rvert)T_{mul}$ $\approx 10,066,329,600$	$15T_{mul}$ $\approx 13,410$	$2T_{mul}$ $\approx 1,788$	$2T_S$ $\approx 2,412$

Table 1: Performance Comparison

Acknowledgments

This work was partially supported by the National Science and Technology Council of Taiwan, under grants 113-2634-F-004-001-MBK, 113-2221-E-004-012-. This research is grateful for the support of "The National Defense Science and Technology Academic Collaborative Research Project in 2025."

	[17]	[20]	[15]	[14]	Ours
Flexible Aut.	Y	Y	Y	Y	Y
Key-Escrow Freeness	Y	N	Y	N	Y
Mathematical Structure	Pairing Groups	Lattices	CDH Groups	TDL Groups	DDH Groups
OW	ROM	STD	ROM	ROM	STD
IND	ROM	STD	ROM	ROM	STD

Table 2: Property Comparison

References

[1] FrodoKEM — frodokem.org. https://frodokem.org/. [Accessed 14-04-2025].

[2] Fahiem Altaf and Soumyadev Maity. PLHAS: Privacy-preserving localized hybrid authentication scheme for large scale vehicular ad hoc networks. *Vehicular Communications*, 30:100347, 2021.

[3] Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. The SPHINCS+ signature framework. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2129–2146, New York, NY, USA, 2019. Association for Computing Machinery.

[4] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, July 2004.

[5] Yu-Chi Chen, Xin Xie, Hung-Yu Tsao, and Raylin Tso. Public key encryption with filtered equality test revisited. *Designs, Codes and Cryptography*, 89(10):2357–2372, 2021.

[6] David Cooper et al. Stateless hash-based digital signature standard, 2024.

[7] Morris J. Dworkin. SHA-3 standard: Permutation-based hash and extendable-output functions, 2015-07-01 04:07:00 2015.

[8] Aurore Guillevic. Pairing-firendly curves, 2021. https://members.loria.fr/AGuillevic/pairing-friendly-curves/.

[9] Thomas Hanson, Qian Wang, Santosh Ghosh, Fernando Virdia, Anne Reinders, and Manoj R. Sastry. Optimization for SPHINCS+ using intel secure hash algorithm extensions. Cryptology ePrint Archive, Paper 2022/1726, 2022.

[10] Kaibin Huang, Raylin Tso, and Yu-Chi Chen. Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption. *Journal of Computer and System Sciences*, 89:400–409, 2017.

[11] Kaibin Huang, Raylin Tso, Yu-Chi Chen, Sk Md Mizanur Rahman, Ahmad Almogren, and Atif Alamri. PKE-AET: public key encryption with authorized equality test. *The Computer Journal*, 58(10):2686–2697, 2015.

[12] Tao Jiang, Xiaofeng Chen, Qianhong Wu, Jianfeng Ma, Willy Susilo, and Wenjing Lou. Secure and efficient cloud data deduplication with randomized tag. *IEEE transactions on information forensics and security*, 12(3):532–543, 2016.

[13] Hyung Tae Lee, San Ling, Jae Hong Seo, Huaxiong Wang, and Taek-Young Youn. Public key encryption with equality test in the standard model. *Information Sciences*, 516:89–108, 2020.

[14] Na Li. Efficient equality test on identity-based ciphertexts supporting flexible authorization. *Entropy*, 25(2):362, 2023.

[15] Xi-Jun Lin, Lin Sun, Haipeng Qu, and Xiaoshuai Zhang. Public key encryption supporting equality test and flexible authorization without bilinear pairings. *Computer Communications*, 170:190–199, 2021.

[16] Yunhao Ling. Tightly secure public key encryption with equality test in setting with adaptive corruptions. *IEEE Access*, 12:115268–115276, 2024.

[17] Sha Ma, Qiong Huang, Mingwu Zhang, and Bo Yang. Efficient public key encryption with equality test supporting flexible authorization. *IEEE Transactions on Information Forensics and Security*, 10(3):458–470, 2014.

[18] Ueli Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In Moni Naor, editor, *Theory of Cryptography*, pages 21–39, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[19] Peter L Montgomery. Modular multiplication without trial division. *Mathematics of computation*, 44(170):519–521, 1985.

[20] Giang Linh Duc Nguyen, Willy Susilo, Dung Hoang Duong, HuyQuoc Le, and Fuchun Guo. Lattice-based IBE with equality test supporting flexible authorization in the standard model. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *Progress in Cryptology – INDOCRYPT 2020*, pages 624–643, Cham, 2020. Springer International Publishing.

[21] Priteshkumar Prajapati and Parth Shah. A review on secure data deduplication: Cloud storage security issue. *Journal of King Saud University-Computer and Information Sciences*, 34(7):3996–4007, 2022.

[22] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[23] Ali Sentürk and Mustafa Gok. A fast modular multiplication method. In *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 204–207, 2010.

[24] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Paper 2004/332, 2004.

[25] Guomin Yang, Chik How Tan, Qiong Huang, and Duncan S Wong. Probabilistic public key encryption with equality test. In *Topics in Cryptology-CT-RSA 2010: The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings*, pages 119–131. Springer, 2010.

[26] Xinying Yang, Cong Yue, Wenhui Zhang, Yang Liu, Beng Chin Ooi, and Jianjun Chen. Secudb: An in-enclave privacy-preserving and tamper-resistant relational database. *Proceedings of the VLDB Endowment*, 17(12):3906–3919, 2024.

[27] Jingze Yu, Wenting Shen, and Xi Zhang. Cloud storage auditing and data sharing with data deduplication and private information protection for cloud-based emr. *Computers & Security*, 144:103932, 2024.

[28] Jingqi Zhang, Zhiming Chen, Mingzhi Ma, Rongkun Jiang, An Wang, Weijiang Wang, and Hua Dang. High-performance elliptic curve scalar multiplication architecture based on interleaved mechanism. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 33(3):757–770, 2025.

[29] Shunsheng Zhang, Youwen Zhu, and Ao Zeng. Collusion-resilient privacy-preserving database fingerprinting. *IEEE Transactions on Information Forensics and Security*, 19:8306–8321, 2024.