entropy

MDPI

Article

Identity-Based Matchmaking Encryption with Equality Test

Zhen Yan, Xijun Lin, Xiaoshuai Zhang, Jianliang Xu and Haipeng Qu *

College of Computer Science and Technology, Ocean University of China, Qingdao 266100, China; yanzhen6751@stu.ouc.edu.cn (Z.Y.); linxj77@ouc.edu.cn (X.L.); x.zhang@ouc.edu.cn (X.Z.); xjl9898@ouc.edu.cn (J.X.)

* Correspondence: quhaipeng@ouc.edu.cn

Abstract: The identity-based encryption with equality test (IBEET) has become a hot research topic in cloud computing as it provides an equality test for ciphertexts generated under different identities while preserving the confidentiality. Subsequently, for the sake of the confidentiality and authenticity of the data, the identity-based signcryption with equality test (IBSC-ET) has been put forward. Nevertheless, the existing schemes do not consider the anonymity of the sender and the receiver, which leads to the potential leakage of sensitive personal information. How to ensure confidentiality, authenticity, and anonymity in the IBEET setting remains a significant challenge. In this paper, we put forward the concept of the identity-based matchmaking encryption with equality test (IBME-ET) to address this issue. We formalized the system model, the definition, and the security models of the IBME-ET and, then, put forward a concrete scheme. Furthermore, our scheme was confirmed to be secure and practical by proving its security and evaluating its performance.

Keywords: matchmaking encryption; equality test; confidentiality; authenticity; anonymity

1. Introduction

The swift progress in cloud computing featured by the outsourcing of data to the cloud has given rise to a growing trend among organizations and individuals, enabling entities to benefit from the ultra-large capacity and calculating services provided by cloud providers. The maintenance of data confidentiality is a fundamental security requirement of cloud storage, which is generally achieved by employing existing cryptographic mechanisms. Nonetheless, how to perform efficient searches on ciphertexts is a practical problem. In order to protect data confidentiality and, meanwhile, support privacy-preserving keyword searching on ciphertexts, public key encryption with keyword search (PEKS) has been presented [1]. Nevertheless, PEKS is limited to searching on ciphertexts generated under a single public key, rendering it unsuitable for cloud storage scenarios involving multiple users.

To provide privacy-preserving equality searching on ciphertexts encrypted under distinct public keys without losing the data confidentiality, Yang et al. [2] put forward an extension of PEKS known as the public key encryption with equality test (PKEET). However, in Yang et al.'s construction, anyone can conduct the equality test without authorization, which infringes on the data owner's privacy. Hence, the authorization mechanism was introduced into the PKEET to guarantee that no one except the data owner can enable the cloud server to test its ciphertexts with the others'.

Subsequently, Ma [3] proposed the identity-based encryption with equality test (IBEET) to eliminate the certificate management problem of the PKEET. In this primitive, the identities of the sender and receiver were exploited to denote the public keys, eliminating the need for certificate management. Owing to the equality test function, the IBEET has been applied in various practical applications, such as personal health record (PHR) systems [4,5] and Internet of Vehicles (IoV) road monitoring [6].

Citation: Yan, Z.; Lin, X.; Zhang, X.; Xu, J.; Qu, H. Identity-Based Matchmaking Encryption with Equality Test. *Entropy* **2024**, *26*, 74. https://doi.org/10.3390/e26010074

Academic Editor: Jaesung Lee

Received: 19 November 2023
Revised: 31 December 2023
Accepted: 8 January 2024
Published: 15 January 2024

Ensuring the authenticity of data is another fundamental security requirement of cloud storage. For the sake of the confidentiality and authenticity of data while supporting the privacy-preserving equality test for ciphertexts generated from different identities, Xiong et al. [7] presented the identity-based signcryption with equality test (IBSC-ET). Afterwards, several related signcryption schemes supporting the equality test have been conceived of. Nevertheless, the existing studies have not considered the anonymity of the sender and the receiver, which leads to the potential leakage of sensitive personal information.

1.1. Motivation

As depicted in Figure 1, in a PHR system, the patients' PHRs contain as much relevant health data as possible from various healthcare providers over their lifetime. To ensure patients' privacy, it is essential to store the health data in the cloud in ciphertext form. To find patients having similar illnesses, a patient (e.g., Alice or Bob) can authorize the cloud server to compare his/her ciphertexts sent by a specified healthcare provider with the others' ciphertexts, so that the patients can help each other by sharing their experiences or mental processes.

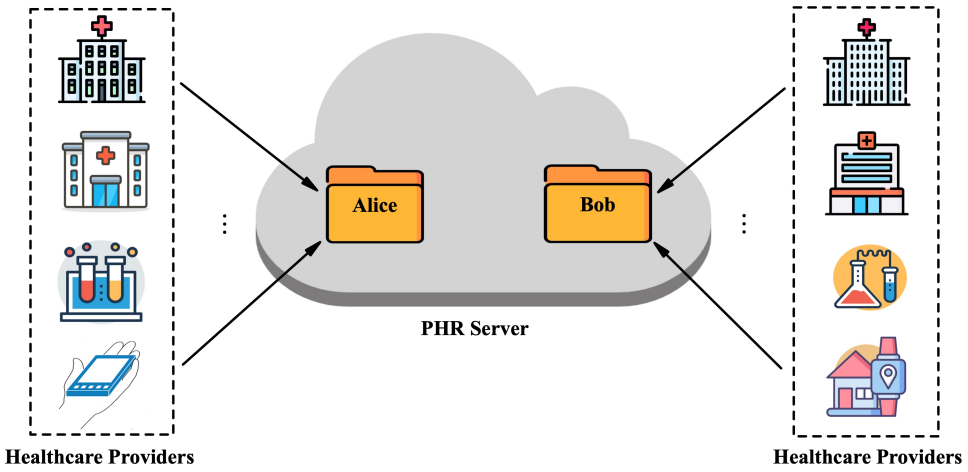

Figure 1. PHR system model.

However, by employing the existing signcryption schemes with equality test (to guarantee the confidentiality and authenticity of health data while supporting the privacy-preserving equality test on ciphertexts), the patients are unable to prevent sensitive personal information from being leaked to the cloud server. That is because the existing schemes do not consider the anonymity of the sender and receiver of the ciphertext. Consequently, the cloud server can know the healthcare provider of the ciphertext, e.g., MD Anderson Cancer Center. Likewise, from the ciphertext and the authorization trapdoor, the cloud server can learn whose identity the ciphertext is encrypted under, namely who is the receiver of the ciphertext, in this way to identify the patient associated with the ciphertext. Obviously, this seriously infringes upon the patient's privacy.

Hence, during the equality testing procedure, there are three security aspects that should be guaranteed against the cloud server:

1. **Confidentiality:** The cloud server has no knowledge about the health data concealed in the ciphertext.
2. **Authenticity:** The cloud server is unable to fake any legitimate ciphertext pertaining to the sender and the receiver.
3. **Anonymity:** The cloud server has no knowledge about the identities of the sender and the receiver concealed in the ciphertext.

Therefore, we propose a new primitive, which not only offers the confidentiality, authenticity, and anonymity of data stored in the cloud, but also provides equality test

functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data.

1.2. Related Works

Search on ciphertexts: Searchable encryption (SE) [8] was put forward to offer secure search functionality over ciphertexts encrypted under single public key. There are two categories of SE: public key encryption with keyword search (PEKS) [1,9,10] and symmetric searchable encryption (SSE) [11,12]. PEKS was conceived of by Boneh et al. [1] to support keyword searching over ciphertexts in public key settings by using the corresponding trapdoors without retrieving messages. After that, a variety of PEKS schemes have been presented for enhanced functionalities and different application requirements [9,10]. However, SE cannot offer equality test functionality for ciphertexts generated under different identities, which differs from our proposal.

Equality test on ciphertexts: The primitive of the PKEET was put forward to verify whether the identical message is concealed in two ciphertexts, where the ciphertexts may be encrypted under distinct public keys [2]. Then, the authorization mechanisms were introduced into the PKEET, and a series of PKEET schemes supporting various authorizations were proposed [13,14]. Ma [3] first introduced the primitive of the IBEET, to eliminate the certificate management problem of the traditional PKEET. A semi-generic IBEET scheme was conceived of by Lee et al. [15] to achieve CCA security. Then, several IBEET schemes supporting various authorizations were introduced [16,17]. Although the above schemes offer equality test functionality while preserving the confidentiality, the data authenticity is not guaranteed. To address this challenge, Xiong et al. [7] established the notion of the IBSC-ET by combining identity-based signcryption (IBSC) [18] and the IBEET. Afterwards, several signcryption schemes with equality test functionality for heterogeneous systems were proposed [19–21]. However, the existing studies have not considered the anonymity of the sender and the receiver, which leads to the potential leakage of sensitive personal information, which differs from our proposal.

Identity-based matchmaking encryption: In CRYPTO 2019, Ateniese et al. [22] put forward the primitive of identity-based matching encryption (IB-ME) to logically ensure the confidentiality, authenticity, and anonymity of data in one step. The guarantee of IB-ME is as follows: the recipient obtains the message when the match happens (both parties' identities match the identity specified by the other party); in case the match does not happen, no information is disclosed other than the fact of the mismatch. Then, by extending IB-ME, a secure access control scheme was conceived of by Xu et al. [23] for cloud–fog computing, and a secure access control scheme was suggested by Sun et al. [24] for cloud-enabled industrial IoT healthcare systems. Chen et al. [25] suggested an IB-ME scheme on the basis of standard assumptions. Wu et al. [26] conceived of a Fuzzy IB-ME scheme. Yan et al. [27] conceived of an IB-ME scheme supporting proxy decryption. Sun et al. [28] suggested an IB-ME scheme supporting a broadcast mechanism. However, although IB-ME can ensure the confidentiality, authenticity, and anonymity of data, all of these related schemes cannot offer equality test functionality for ciphertexts without losing the confidentiality, authenticity, and anonymity of the data, which differs from our proposal.

1.3. Contributions

We emphasize here again that the existing cryptographic schemes with the equality test do not consider the anonymity of the sender and the receiver, which leads to the potential leakage problem of sensitive personal information. Hence, we put forward a novel primitive, called the identity-based matchmaking encryption with equality test (IBME-ET), by combining IB-ME and the IBEET. This primitive not only offers the confidentiality, authenticity, and anonymity of data stored in the cloud, but also provides equality test functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data.

Our proposed IBME-ET can advance the anonymity of existing applications. For example, in a PHR system [4,5], the patient can permit the cloud server to compare his/her encrypted health data sent by a specified healthcare provider with the others', in this way to make friends with the patients having a similar illness. Our proposal can simplify the leakage problem of the real identities of the healthcare provider and the patient, which exists in current cryptographic schemes with the equality test, thereby guaranteeing the confidentiality, authenticity, and anonymity of the patients' health data.

The equality testing process in the IBME-ET can be succinctly outlined as follows: Let $C_{(\sigma_A, rcv_A)}$ denote a ciphertext generated on $(ek_{\sigma_A}, rcv_A, m_A)$ and $C_{(\sigma_B, rcv_B)}$ denote a ciphertext generated on $(ek_{\sigma_B}, rcv_B, m_B)$, where ek_{σ_A} and ek_{σ_B} are the encryption keys of the senders with identities σ_A and σ_B and rcv_A and rcv_B are the identities of the specified receivers, respectively. Furthermore, let $td_{(snd_A, \rho_A)}$ be a trapdoor generated on (snd_A, dk_{ρ_A}) and $td_{(snd_B, \rho_B)}$ be a trapdoor generated on (snd_B, dk_{ρ_B}), where snd_A and snd_B are the identities of the specified senders and dk_{ρ_A} and dk_{ρ_B} are the decryption keys of the receivers with identities ρ_A and ρ_B, respectively. Given $(C_{(\sigma_A, rcv_A)}, td_{(snd_A, \rho_A)})$ and $(C_{(\sigma_B, rcv_B)}, td_{(snd_B, \rho_B)})$, two conditions are involved:

- Match (i.e., $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge m_A = m_B$): the cloud server returns 1, and no further information is revealed other than the fact that the match happened, that is the cloud server learns neither the messages $m_A = m_B$ nor the identities $\sigma_A = snd_A, rcv_A = \rho_A, \sigma_B = snd_B, rcv_B = \rho_B$.
- Mismatch (i.e., $\sigma_A \neq snd_A \vee rcv_A \neq \rho_A \vee \sigma_B \neq snd_B \vee rcv_B \neq \rho_B \vee m_A \neq m_B$): the cloud server returns 0, and no further information is revealed other than the fact of the mismatch, that is the cloud server learns neither the messages m_A, m_B nor the identities $\sigma_A, snd_A, rcv_A, \rho_A, \sigma_B, snd_B, rcv_B, \rho_B$.

The principal contributions can be succinctly outlined as follows:

1. We present the notion of the IBME-ET, which not only offers the confidentiality, authenticity, and anonymity of data stored in the cloud, but also provides equality test functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data.
2. We put forward the system model and definition of the IBME-ET. With respect to the confidentiality, authenticity, and anonymity, we formulated four security models for the IBME-ET by taking four types of adversaries into account.
3. We constructed a concrete IBME-ET scheme on the basis of the BDH assumption and the Gap-BDH assumption. Our scheme was confirmed to be secure and practical by proving its security and evaluating its performance.

1.4. Organization

In general: Section 2 introduces the preliminaries while Section 3 presents IBME-ET by displaying its system, definition and four security models. Sections 4 and 5, respectively, focus on the detailed scheme and analysis of security. Then, Section 6 focuses on performance evaluation, Section 7 arrives at a conclusion.

2. Preliminaries

2.1. Asymmetric Bilinear Groups

$\mathbb{G}, \hat{\mathbb{G}}$, and $\mathbb{G}_T$ indicate three multiplicative cyclic groups with prime order q. g and $\hat{g}$ are the generators of $\mathbb{G}$ and $\hat{\mathbb{G}}$, respectively. An asymmetric bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ includes the following characteristics:

1. *Bilinearity:* $\forall x \in \mathbb{G}, \forall y \in \hat{\mathbb{G}}$ and $\forall u, v \in \mathbb{Z}_q^*, e(x^u, y^v) = e(x, y)^{uv}$.
2. *Non-degeneracy:* $\exists g \in \mathbb{G}, \hat{g} \in \hat{\mathbb{G}}, e(g, \hat{g}) \neq 1$.

Note that the group operations and asymmetric bilinear map e can be computed efficiently. However, if no efficiently computable isomorphisms are found between $\mathbb{G}$ and $\hat{\mathbb{G}}$, then $\mathbb{G}, \hat{\mathbb{G}}$ and $\mathbb{G}_T$ do not possess efficiently computable isomorphisms.

2.2. Assumptions

1. *Bilinear Diffie–Hellman (BDH) assumption:* When a tuple $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b) \in \mathbb{G}^3 \times \hat{\mathbb{G}}^3$ is given, no PPT algorithm $\mathcal{A}$ calculates $e(g, \hat{g})^{abc} \in \mathbb{G}_T$ with non-negligible advantage. Define $\mathcal{A}$'s advantage as

$$Adv^{\mathcal{A}}_{BDH}(\lambda) = \Pr[\mathcal{A}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b) = e(g, \hat{g})^{abc}].$$

2. *Gap-bilinear Diffie–Hellman (Gap-BDH) assumption:* When a tuple $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b) \in \mathbb{G}^3 \times \hat{\mathbb{G}}^3$ is given, even with the decision BDH oracle $\mathcal{O}_{\mathcal{DBDH}}$, no PPT algorithm $\mathcal{A}$ calculates $e(g, \hat{g})^{abc} \in \mathbb{G}_T$ with non-negligible advantage [29]. Tuples of the form $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, e(g, \hat{g})^{abc})$ are known as "BDH tuples". With $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, T)$, $\mathcal{O}_{\mathcal{DBDH}}$ is able to check $T = e(g, \hat{g})^{abc}$ or not. $\mathcal{O}_{\mathcal{DBDH}}$ outputs 1 when $T = e(g, \hat{g})^{abc}$; otherwise, $\mathcal{O}_{\mathcal{DBDH}}$ outputs 0. Define $\mathcal{A}$'s advantage as

$$Adv^{\mathcal{A}}_{Gap\text{-}BDH}(\lambda) = \Pr[\mathcal{A}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \mathcal{O}_{\mathcal{DBDH}}) = e(g, \hat{g})^{abc}].$$

3. Definitions of IBME-ET

3.1. System Model

In Figure 2, our proposed IBME-ET comprises four distinct entities.

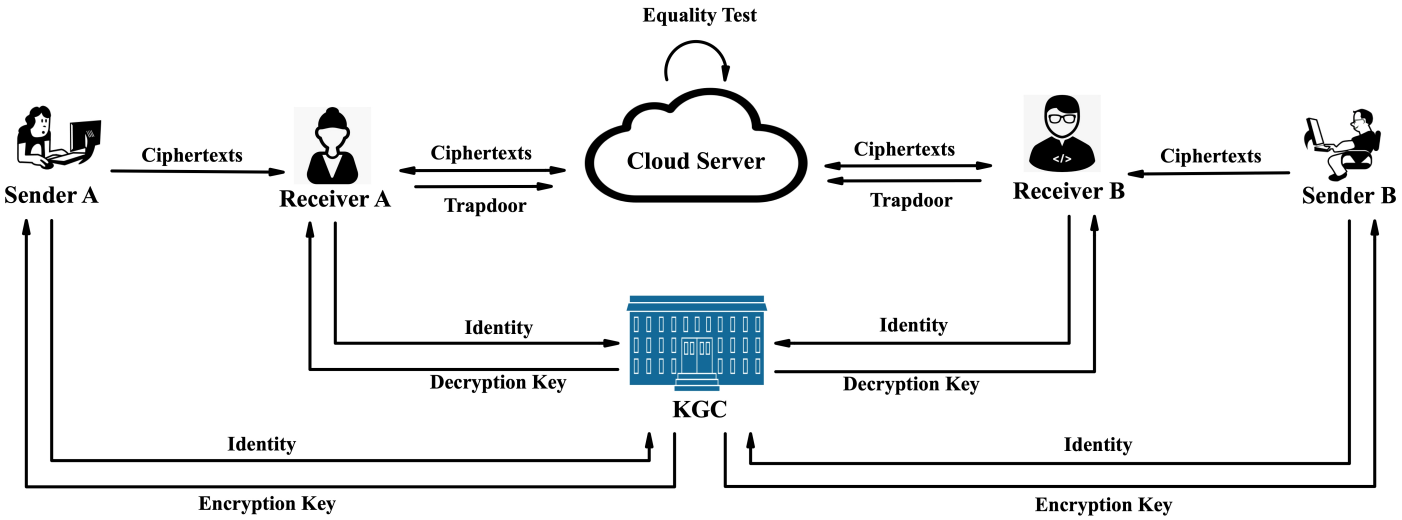

Figure 2. IBME-ET system model.

- *KGC:* This entity's responsibility is to securely generate and distribute encryption keys and decryption keys.
- *Sender:* This entity's responsibility is to generate ciphertexts, ensuring the confidentiality, authenticity, and anonymity of the data.
- *Receiver:* This entity is responsible for collecting and outsourcing ciphertexts from potential senders secretly. It permits the cloud server to test ciphertexts sent by a specific sender without compromising the confidentiality, authenticity, and anonymity of the data.
- *Cloud server:* This entity's responsibility is to store the ciphertexts and perform equality tests based on the receivers' authorizations.

Our workflow is succinctly outlined as follows:

1. The KGC utilizes the algorithm *SKGen* to calculate the encryption key ek_σ in accordance with the identity of the sender σ and securely delivers this to the sender. Similarly, the KGC utilizes the algorithm *RKGen* to calculate the decryption key dk_ρ in accordance with the identity of the receiver ρ and securely delivers this to the receiver.

2. A sender identified as σ executes the algorithm *Enc* to conceal the message m using encryption key ek_σ along with a target receiver's identity rcv, delivering it to the receiver with the ciphertext $C_{(\sigma,rcv)}$.
3. A receiver identified as ρ executes the algorithm *Decc* to decrypt the ciphertexts by employing the receiver's decryption key dk_ρ and the identity of the target sender snd, delivering the desirable ciphertexts to the cloud server. Specifically, given $C_{(\sigma,rcv)}$, dk_ρ, and snd, the guarantee in the decryption procedure is as follows:
   - Match (i.e., $\sigma = snd \wedge \rho = rcv$): the message m is obtained by the receiver.
   - Mismatch (i.e.,$\sigma \neq snd \vee \rho \neq rcv$): the receiver obtains neither the message m nor the identities σ, rcv.
4. To test the ciphertexts offered by a target sender, the receiver identified as ρ executes the algorithm *Auth* to calculate a trapdoor $td_{(snd,\rho)}$ with the identity of the target sender snd and its decryption key dk_ρ and delivers the trapdoor to the cloud server.
5. Utilizing the receivers' trapdoors, the cloud server executes the algorithm *Test* to test the ciphertexts sent by the specified senders without learning the messages and identities. Specifically, given $(C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)})$ and $(C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)})$, the guarantee in equality testing procedure is as follows:
   - Match (i.e., $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge m_A = m_B$): the cloud server returns 1, and the cloud server learns neither the messages $m_A = m_B$ nor the identities $\sigma_A = snd_A, rcv_A = \rho_A, \sigma_B = snd_B, rcv_B = \rho_B$.
   - Mismatch (i.e., $\sigma_A \neq snd_A \vee rcv_A \neq \rho_A \vee \sigma_B \neq snd_B \vee rcv_B \neq \rho_B \vee m_A \neq m_B$): the cloud server returns 0, and the cloud server learns neither the messages m_A, m_B nor the identities σ_A, snd_A, rcv_A, ρ_A, σ_B, snd_B, rcv_B, ρ_B.

3.2. IBME-ET Definition

An IBME-ET scheme comprises the subsequent algorithms:

- $Setup(\lambda) \rightarrow (pp, mk)$: The system parameters pp along with the master key mk are answered.
- $SKGen(pp, mk, \sigma) \rightarrow ek_\sigma$: The encryption key ek_σ for the sender identified as σ is answered.
- $RKGen(pp, mk, \rho) \rightarrow dk_\rho$: The decryption key dk_ρ for the receiver identified as ρ is answered.
- $Enc(pp, ek_\sigma, rcv, m) \rightarrow C$: Given the system parameters pp, an encryption key of the sender ek_σ, and an identity of the target receiver rcv along with the message m, the corresponding ciphertext C is answered.
- $Dec(pp, dk_\rho, snd, C) \rightarrow m/\perp$: Given the system parameters pp, a decryption key of the receiver dk_ρ, and an identity of the target sender snd along with the ciphertext C, the corresponding message m is answered or the symbol $\perp$ to signal the failure of the decryption is answered.
- $Auth(pp, snd, dk_\rho) \rightarrow td_{(snd,\rho)}$: Given the system parameters pp and an identity of the target sender snd along with a decryption key of the receiver dk_ρ, the corresponding trapdoor $td_{(snd,\rho)}$ is answered.
- $Test(pp, C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)}, C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)}) \rightarrow 0/1$: Given the system parameters pp, two pairs of ciphertext/trapdoors $(C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)})$ and $(C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)})$, if $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge C_{(\sigma_A,rcv_A)}$ and $C_{(\sigma_B,rcv_B)}$ are generated using the identical message, it answers 1. Otherwise, it answers 0.

Correctness: An IBME-ET scheme is correct when the subsequent conditions are met:

1. When $\sigma = snd \wedge \rho = rcv$, $Dec(pp, dk_\rho, snd, Enc(pp, ek_\sigma, rcv, m)) = m$ always holds.
2. Let $C_{(\sigma_A,rcv_A)} = Enc(pp, ek_{\sigma_A}, rcv_A, m_A)$, $C_{(\sigma_B,rcv_B)} = Enc(pp, ek_{\sigma_B}, rcv_B, m_B)$, $td_{(snd_A,\rho_A)} = Auth(pp, snd_A, dk_{\rho_A})$, and $td_{(snd_B,\rho_B)} = Auth(pp, snd_B, dk_{\rho_B})$. If $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge m_A = m_B$, $Test(pp, C_{(\sigma_A,rcv_A)},$

$td_{(snd_A,\rho_A)}, C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)}) = 1$; otherwise, $\Pr[Test(pp, C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)}, C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)}) = 1]$ is negligible.

3.3. Security Definitions

With respect to the confidentiality, authenticity, and anonymity of the IBME-ET, it is crucial to consider four distinct types of adversaries:

- *Type-I adversary* $\mathcal{A}_1$: Without the trapdoor and decryption key of the receiver, $\mathcal{A}_1$ is unable to determine which message the challenge ciphertext is computed from. For $\mathcal{A}_1$, define the security model IND-ID-CCA.
- *Type-II adversary* $\mathcal{A}_2$: Without the decryption key of the receiver, $\mathcal{A}_2$ is unable to obtain the message concealed in the challenge ciphertext. For $\mathcal{A}_2$, define the security model OW-ID-CCA.
- *Type-III adversary* $\mathcal{A}_3$: Without the decryption key of the receiver and the encryption key of the sender, $\mathcal{A}_3$ is unable to determine the corresponding sender and receiver, even if $\mathcal{A}_3$ has the trapdoor. For $\mathcal{A}_3$, define the security model ANON-ID-CCA.
- *Type-IV adversary* $\mathcal{A}_4$: Without the decryption key of the receiver and the encryption key of the sender, $\mathcal{A}_4$ is unable to fake any legitimate ciphertext delivered by the sender to the receiver, even if $\mathcal{A}_4$ has the trapdoor. For $\mathcal{A}_4$, define the security model sUF-ID-CMA.

Let $\mathcal{C}$ be the challenger. We have the following oracles:

- $\mathcal{O}_{SKGen}(\sigma_i)$: Once the identity of the sender σ_i is received, $\mathcal{C}$ answers the encryption key ek_{σ_i}.
- $\mathcal{O}_{RKGen}(\rho_j)$: Once the identity of the receiver ρ_j is received, $\mathcal{C}$ answers the decryption key dk_{ρ_j}.
- $\mathcal{O}_{Enc}(\sigma_i, rcv, m)$: Once the identity of the sender σ_i, the identity of the target receiver rcv, and a message m are received, $\mathcal{C}$ answers the result of $Enc(pp, ek_{\sigma_i}, rcv, m)$.
- $\mathcal{O}_{Dec}(\rho_j, snd, C)$: Once the identity of the receiver ρ_j, the identity of the target sender snd, and a ciphertext C are received, $\mathcal{C}$ answers the result of $Dec(pp, dk_{\rho_j}, snd, C)$.
- $\mathcal{O}_{Auth}(snd, \rho_j)$: Once the identity of the target sender snd and the identity of the receiver ρ_j are received, $\mathcal{C}$ answers the corresponding trapdoor $td_{(snd,\rho_j)} = Auth(pp, snd, dk_{\rho_j})$.

Definition 1 (IND-ID-CCA)**.** *Regarding $\mathcal{A}_1$, the IBME-ET scheme meets IND-ID-CCA security when no PPT $\mathcal{A}_1$ is winning the game below with a non-negligible advantage:*

1. *Setup: $\mathcal{C}$ utilizes the algorithm $Setup$ to calculate the master key mk and the system parameters pp and delivers pp to $\mathcal{A}_1$.*
2. *Phase 1: $\mathcal{A}_1$ can issue queries to the oracles: $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Dec}$.*
3. *Challenge: $\mathcal{A}_1$ sends identities σ^*, rcv^* and equal-length messages m_0^*, m_1^* to $\mathcal{C}$. Subsequently, $\mathcal{C}$ randomly selects $x \in \{0,1\}$ and answers $\mathcal{A}_1$ with the challenge ciphertext $C^* = Enc(pp, ek_{\sigma^*}, rcv^*, m_x^*)$.*
4. *Phase 2: $\mathcal{A}_1$ makes queries like in Phase 1.*
5. *Guess: $\mathcal{A}_1$ answers a guess $x' \in \{0,1\}$ and is winning when $x = x'$. $\mathcal{A}_1$'s advantage is defined as $Adv_{IBME-ET,\mathcal{A}_1}^{IND-ID-CCA}(\lambda) = |Pr[x = x'] - \frac{1}{2}|$.*

In the above game, the constraint is that $\mathcal{A}_1$ cannot ask the following queries: $\mathcal{O}_{RKGen}(rcv^*)$, $\mathcal{O}_{Auth}(\sigma^*, rcv^*)$, $\mathcal{O}_{Dec}(rcv^*, \sigma^*, C^*)$.

Definition 2 (OW-ID-CCA)**.** *Regarding $\mathcal{A}_2$, the IBME-ET scheme meets OW-ID-CCA security when no PPT $\mathcal{A}_2$ is winning the game below with a non-negligible advantage:*

1. *Setup: Same as Definition 1.*
2. *Phase 1: $\mathcal{A}_2$ can issue queries to the oracles: $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Dec}$.*
3. *Challenge: $\mathcal{A}_2$ sends identities σ^*, rcv^* to $\mathcal{C}$. Subsequently, $\mathcal{C}$ randomly chooses a message $m^* \in \{0,1\}^\lambda$ and answers to $\mathcal{A}_2$ with the challenge ciphertext $C^* = Enc(pp, ek_{\sigma^*}, rcv^*, m^*)$.*
4. *Phase 2: $\mathcal{A}_2$ makes queries like in Phase 1.*

5. *Guess: $\mathcal{A}_2$ answers a guess m' and is winning when $m^* = m'$. $\mathcal{A}_2$'s advantage is defined as $Adv^{OW-ID-CCA}_{IBME-ET,\mathcal{A}_2}(\lambda) = Pr[m^* = m']$.*

In the above game, the constraints is that $\mathcal{A}_2$ cannot ask the following queries: $\mathcal{O}_{RKGen}(rcv^*)$, $\mathcal{O}_{Dec}(rcv^*, \sigma^*, C^*)$.

Definition 3 (ANON-ID-CCA)**.** *Regarding $\mathcal{A}_3$, the IBME-ET scheme meets ANON-ID-CCA security when no PPT $\mathcal{A}_3$ is winning the game below with a non-negligible advantage:*

1. *Setup: Same as Definition 1.*
2. *Phase 1: $\mathcal{A}_3$ can issue queries to the oracles: $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Enc}$, $\mathcal{O}_{Dec}$.*
3. *Challenge: $\mathcal{A}_3$ sends identities (snd^*_0, ρ^*_0), (snd^*_1, ρ^*_1) and a message m^* to $\mathcal{C}$. Subsequently, $\mathcal{C}$ randomly chooses $x \in \{0,1\}$ and answers to $\mathcal{A}_3$ with the challenge ciphertext $C^* = Enc(pp, ek_{snd^*_x}, \rho^*_x, m^*)$ and the challenge trapdoor $td_{(snd^*_x, \rho^*_x)} = Auth(pp, snd^*_x, dk_{\rho^*_x})$.*
4. *Phase 2: $\mathcal{A}_3$ makes queries like in Phase 1.*
5. *Guess: $\mathcal{A}_3$ answers a guess $x' \in \{0,1\}$ and is winning when $x = x'$. $\mathcal{A}_3$'s advantage is defined as $Adv^{ANON-ID-CCA}_{IBME-ET,\mathcal{A}_3}(\lambda) = |Pr[x = x'] - \frac{1}{2}|$.*

In the above game, the constraint is that $\mathcal{A}_3$ cannot ask the following queries:

- $\mathcal{O}_{SKGen}(snd^*_0)$, $\mathcal{O}_{SKGen}(snd^*_1)$, $\mathcal{O}_{Enc}(snd^*_0, \rho^*_0, *)$ and $\mathcal{O}_{Enc}(snd^*_1, \rho^*_1, *)$.
- $\mathcal{O}_{RKGen}(\rho^*_0)$, $\mathcal{O}_{RKGen}(\rho^*_1)$, $\mathcal{O}_{Auth}(snd^*_0, \rho^*_0)$ and $\mathcal{O}_{Auth}(snd^*_1, \rho^*_1)$.
- $\mathcal{O}_{Dec}(\rho^*_0, snd^*_0, C^*)$, $\mathcal{O}_{Dec}(\rho^*_1, snd^*_1, C^*)$.

Definition 4 (sUF-ID-CMA)**.** *Regarding $\mathcal{A}_4$, the IBME-ET scheme meets sUF-ID-CMA security when no PPT $\mathcal{A}_4$ is winning the game below with a non-negligible advantage:*

1. *Setup: Same as Definition 1.*
2. *Queries: $\mathcal{A}_4$ can issue queries to the oracles: $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Enc}$, $\mathcal{O}_{Dec}$.*
3. *Forgery: $\mathcal{A}_4$ answers a triple (snd^*, ρ^*, C^*). $\mathcal{A}_4$ is winning when $m^* = Dec(pp, dk_{\rho^*}, snd^*, C^*) \neq \perp$. $\mathcal{A}_4$'s advantage is defined as $Adv^{sUF-ID-CMA}_{IBME-ET,\mathcal{A}_4}(\lambda) = Pr[\mathcal{A}_4 \text{ wins}]$.*

In the above game, the constraint is that $\mathcal{A}_4$ cannot make the following queries: $\mathcal{O}_{SKGen}(snd^*)$ and $\mathcal{O}_{RKGen}(\rho^*)$. Furthermore, C^* cannot be an output of $\mathcal{O}_{Enc}(snd^*, \rho^*, *)$.

4. Our Construction

The IBME-ET scheme is concretely constructed as below:

- *Setup*(λ): The following steps are taken:
  1. Randomly select the generators $g \in \mathbb{G}$ along with $\hat{g} \in \hat{\mathbb{G}}$.
  2. Randomly select numbers $s, \alpha, \beta_0, \beta_1 \in \mathbb{Z}^*_q$, and set $g_1 = g^\alpha$, $f = g^{\beta_0}$, $\hat{f} = \hat{g}^{\beta_0}$, $h = g^{\beta_1}$, $\hat{h} = \hat{g}^{\beta_1}$.
  3. Secure hash functions are defined: $H : \mathbb{G}_T \to \mathbb{Z}^*_q$, $H_1 : \{0,1\}^* \to \mathbb{G}$, $H_2 : \{0,1\}^* \to \hat{\mathbb{G}}$, $H_3 : \{0,1\}^* \to \hat{\mathbb{G}}$, $H_4 : \mathbb{G}_T \to \mathbb{Z}^*_q$, $H_5 : \{0,1\}^{\lambda+l} \to \mathbb{Z}^*_q$, $H_6 : \mathbb{G}^2_T \times \mathbb{G}^3 \to \{0,1\}^{\lambda+l}$, $H_7 : \{0,1\}^\lambda \to \hat{\mathbb{G}}$, and $H_8 : \mathbb{G}_T \to \hat{\mathbb{G}}$.
  4. Return the master key mk along with the system parameters pp, where
  $$mk = (s, \alpha),$$
  $$pp = (\mathcal{G}, g, \hat{g}, g_1, f, h, \hat{f}, \hat{h}, H, H_1, H_2, H_3, H_4, H_5, H_6, H_7, H_8).$$
- *SKGen*(pp, mk, σ): Let $mk = (s, \alpha)$. This algorithm produces the encryption key $ek_\sigma = H_1(\sigma)^s$.
- *RKGen*(pp, mk, ρ): Let $mk = (s, \alpha)$. This algorithm produces the decryption key $dk_\rho = (d_1, d_2, d_3) = (H_3(\rho)^s, H_2(\rho)^\alpha, H_3(\rho)^\alpha)$.
- *Enc*(pp, ek_σ, rcv, m): Let $rcv = \rho$ and $m \in \{0,1\}^\lambda$. The ciphertext $C = (C_0, C_1, C_2, C_3, C_4)$ is calculated as below:

1. Randomly select $r \in \mathbb{Z}_q^*$ and $k \in \{0,1\}^l$, and calculate $R = H_5(m,k)$.
2. Calculate $\eta = e(ek_\sigma, H_3(\rho))$, $\omega_1 = e(g_1, H_2(\rho))^{r \cdot H_4(\eta)}$ and $\omega_2 = e(g_1, H_3(\rho))^{r \cdot H_4(\eta)}$.
3. Calculate the following numbers:

$$\begin{cases} C_0 = g^R, \\ C_1 = g^r, \\ C_2 = (fh^{H(\eta)})^r, \\ C_3 = (m \parallel k) \oplus H_6(\omega_1, \eta, C_0, C_1, C_2), \\ C_4 = H_7(m)^R \cdot H_8(\omega_2). \end{cases}$$

- $Dec(pp, dk_\rho, snd, C)$: Let $dk_\rho = (d_1, d_2, d_3)$, $snd = \sigma$. The following steps are taken:
  1. Calculate $\eta = e(H_1(\sigma), d_1)$, $\omega_1 = e(C_1, d_2^{H_4(\eta)})$ and $\omega_2 = e(C_1, d_3^{H_4(\eta)})$.
  2. Obtain $m' \parallel k'$ by computing $C_3 \oplus H_6(\omega_1, \eta, C_0, C_1, C_2)$.
  3. Calculate $R' = H_5(m', k')$.
  4. If $C_0 = g^{R'}$ and $C_4 = H_7(m')^{R'} \cdot H_8(\omega_2)$ hold, answer m'; otherwise, answer $\perp$.
- $Auth(pp, snd, dk_\rho)$: Let $dk_\rho = (d_1, d_2, d_3)$ and $snd = \sigma$. The following steps are taken:
  1. Randomly select $y \in \mathbb{Z}_q^*$, and calculate $\eta = e(H_1(\sigma), d_1)$.
  2. Return the trapdoor $td_{(snd,\rho)} = (y_1, y_2) = (d_3^{H_4(\eta)}(\hat{f}\hat{h}^{H(\eta)})^y, \hat{g}^y)$.
- $Test(pp, C_{(\sigma_A, rcv_A)}, td_{(snd_A,\rho_A)}, C_{(\sigma_B, rcv_B)}, td_{(snd_B,\rho_B)})$: Let $C_{(\sigma_A, rcv_A)} = (C_{\sigma_A, rcv_A,0}, C_{\sigma_A, rcv_A,1}, C_{\sigma_A, rcv_A,2}, C_{\sigma_A, rcv_A,3}, C_{\sigma_A, rcv_A,4})$, $td_{(snd_A,\rho_A)} = (y_{snd_A,\rho_A,1}, y_{snd_A,\rho_A,2})$, $C_{(\sigma_B, rcv_B)} = (C_{\sigma_B, rcv_B,0}, C_{\sigma_B, rcv_B,1}, C_{\sigma_B, rcv_B,2}, C_{\sigma_B, rcv_B,3}, C_{\sigma_B, rcv_B,4})$ and $td_{(snd_B,\rho_B)}) = (y_{snd_B,\rho_B,1}, y_{snd_B,\rho_B,2})$. The following steps are taken:
  1. Calculate

$$\omega_{A,2} = e(C_{\sigma_A, rcv_A,1}, y_{snd_A,\rho_A,1}) / e(C_{\sigma_A, rcv_A,2}, y_{snd_A,\rho_A,2}),$$

$$\omega_{B,2} = e(C_{\sigma_B, rcv_B,1}, y_{snd_B,\rho_B,1}) / e(C_{\sigma_B, rcv_B,2}, y_{snd_B,\rho_B,2}).$$

  2. Calculate

$$K_A = C_{\sigma_A, rcv_A,4} / H_8(\omega_{A,2}),$$

$$K_B = C_{\sigma_B, rcv_B,4} / H_8(\omega_{B,2}).$$

  3. Check whether $e(C_{\sigma_A, rcv_A,0}, K_B) = e(C_{\sigma_B, rcv_B,0}, K_A)$ holds. When it holds, answer 1 or 0 otherwise.

Correctness: The proposed scheme is correct in accordance with the correctness definition:

1. Regarding Condition 1, when $\sigma = snd$ and $\rho = rcv$, we have

$$\eta = e(ek_\sigma, H_3(\rho)) = e(H_1(\sigma), H_3(\rho))^s = e(H_1(\sigma), d_1),$$
$$\omega_1 = e(g_1, H_2(\rho))^{r \cdot H_4(\eta)} = e(g, H_2(\rho))^{r\alpha \cdot H_4(\eta)} = e(C_1, d_2^{H_4(\eta)}),$$
$$C_3 \oplus H_6(\omega_1, \eta, C_0, C_1, C_2) = (m \parallel k) \oplus H_6(\omega_1, \eta, C_0, C_1, C_2) \oplus H_6(\omega_1, \eta, C_0, C_1, C_2) = m \parallel k.$$

   Thus, when $\sigma = snd$ and $\rho = rcv$, $Dec(pp, dk_\rho, snd, Enc(pp, ek_\sigma, rcv, m)) = m$ always holds.
2. Regarding Condition 2, if $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge m_A = m_B$, we have

$$\begin{aligned}
\frac{e(C_{\sigma_A,\rho_A,1}, y_{\sigma_A,\rho_A,1})}{e(C_{\sigma_A,\rho_A,2}, y_{\sigma_A,\rho_A,2})} &= \frac{e(g^{r_A}, d_{A,3}^{H_4(\eta_A)}(\hat{f}\hat{h}^{H(\eta_A)})^{y_A})}{e((fh^{H(\eta_A)})^{r_A}, \hat{g}^{y_A})} = \frac{e(g^{r_A}, d_{A,3}^{H_4(\eta_A)}) \cdot e(g, \hat{f}\hat{h}^{H(\eta_A)})^{r_A y_A}}{e(fh^{H(\eta_A)}, \hat{g})^{r_A y_A}} \\
&= \frac{e(g^{r_A}, d_{A,3}^{H_4(\eta_A)}) \cdot e(g, \hat{g}^{\beta_0+\beta_1 H(\eta_A)})^{r_A y_A}}{e(g^{\beta_0+\beta_1 H(\eta_A)}, \hat{g})^{r_A y_A}} = e(g^{r_A}, d_{A,3}^{H_4(\eta_A)}) \\
&= e(g, H_3(\rho_A))^{r_A \alpha \cdot H_4(\eta_A)} = e(g_1, H_3(\rho_A))^{r_A \cdot H_4(\eta_A)} = \omega_{A,2}, \\
\frac{e(C_{\sigma_B,\rho_B,1}, y_{\sigma_B,\rho_B,1})}{e(C_{\sigma_B,\rho_B,2}, y_{\sigma_B,\rho_B,2})} &= \frac{e(g^{r_B}, d_{B,3}^{H_4(\eta_B)}(\hat{f}\hat{h}^{H(\eta_B)})^{y_B})}{e((fh^{H(\eta_B)})^{r_B}, \hat{g}^{y_B})} = \frac{e(g^{r_B}, d_{B,3}^{H_4(\eta_B)}) \cdot e(g, \hat{f}\hat{h}^{H(\eta_B)})^{r_B y_B}}{e(fh^{H(\eta_B)}, \hat{g})^{r_B y_B}} \\
&= \frac{e(g^{r_B}, d_{B,3}^{H_4(\eta_B)}) \cdot e(g, \hat{g}^{\beta_0+\beta_1 H(\eta_B)})^{r_B y_B}}{e(g^{\beta_0+\beta_1 H(\eta_B)}, \hat{g})^{r_B y_B}} = e(g^{r_B}, d_{B,3}^{H_4(\eta_B)}) \\
&= e(g, H_3(\rho_B))^{r_B \alpha \cdot H_4(\eta_B)} = e(g_1, H_3(\rho_B))^{r_B \cdot H_4(\eta_B)} = \omega_{B,2}.
\end{aligned}$$

$$\begin{aligned}
K_A &= \frac{C_{\sigma_A,\rho_A,4}}{H_8(\omega_{A,2})} = \frac{H_7(m_A)^{R_A} \cdot H_8(\omega_{A,2})}{H_8(\omega_{A,2})} = H_7(m_A)^{R_A}, \\
K_B &= \frac{C_{\sigma_B,\rho_B,4}}{H_8(\omega_{B,2})} = \frac{H_7(m_B)^{R_B} \cdot H_8(\omega_{B,2})}{H_8(\omega_{B,2})} = H_7(m_B)^{R_B}, \\
e(C_{\sigma_A,\rho_A,0}, K_B) &= e(g^{R_A}, H_7(M_B)^{R_B}) = e(g, H_7(M_B))^{R_A R_B}, \\
e(C_{\sigma_B,\rho_B,0}, K_A) &= e(g^{R_B}, H_7(M_A)^{R_A}) = e(g, H_7(M_A))^{R_A R_B}.
\end{aligned}$$

If $\sigma_A = snd_A \wedge rcv_A = \rho_A \wedge \sigma_B = snd_B \wedge rcv_B = \rho_B \wedge m_A = m_B$, then $e(C_{\sigma_A,\rho_A,0}, K_B) = e(C_{\sigma_B,\rho_B,0}, K_A)$, so $Test(pp, C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)}, C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)}) = 1$; otherwise, $\Pr[Test(pp, C_{(\sigma_A,rcv_A)}, td_{(snd_A,\rho_A)}, C_{(\sigma_B,rcv_B)}, td_{(snd_B,\rho_B)}) = 1]$ is negligible due to the hash functions H_7 and H_8 being collision-resistant.

5. Security Analysis

In the random oracle model, we used the method of proof by contradiction to show that if the BDH assumption and Gap-BDH assumption introduced in the preliminaries (see Section 2) hold, and our proposed IBME-ET scheme can meet confidentiality, authenticity, and anonymity in cryptography [30–32].

According to our IBME-ET scheme, given the ciphertext C, we have the following observations:

- To reveal the message m, it is necessary to calculate $\omega_1 = e(g_1, H_2(\rho))^{r \cdot H_4(\eta)}$.
- To obtain $H_7(m)^R$, which is used for the equality test, it is necessary to calculate $\omega_2 = e(g_1, H_3(\rho))^{r \cdot H_4(\eta)}$.
- To distinguish the identities of the sender and the receiver concealed in the ciphertext, it is necessary to calculate $\eta = e(ek_\sigma, H_3(\rho)) = e(H_1(\sigma), H_3(\rho))^s$.
- To fake any legitimate ciphertext pertaining to the sender σ and the receiver ρ, it is necessary to calculate $\eta = e(ek_\sigma, H_3(\rho)) = e(H_1(\sigma), H_3(\rho))^s$.

Note that, regarding to the confidentiality, anonymity, and authenticity of the IBME-ET, four security models are defined by considering four distinct types of adversaries (see Section 3.3). The security proof of our scheme can be outlined as follows:

As for the confidentiality, we first used the BDH assumption to prove that our proposal meets IND-ID-CCA security regarding the Type-I adversary $\mathcal{A}_1$. Given a BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b)$, we generated a simulated scheme $\mathcal{B}$ and interacted with $\mathcal{A}_1$ by following the IND-ID-CCA security model defined in Section 3.3. $\mathcal{B}$ simulates the oracles $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, and $\mathcal{O}_{Dec}$ to answer $\mathcal{A}_1$'s queries and preserves the L_H and $L_{H_i}(i = 1, 2, 3, 5, 6, 7, 8)$ lists to simulate the random oracles $\mathcal{O}_H$ and $\mathcal{O}_{H_i}(i = 1, 2, 3, 5, 6, 7, 8)$. In the challenge phase, $\mathcal{A}_1$ sends identities σ^*, rcv^* and equal-length messages m_0^*, m_1^* to $\mathcal{B}$. Let $rcv^* = \rho^*$. $\mathcal{B}$ randomly selects $x \in \{0, 1\}$ and answers the challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*) = Enc(pp, ek_{\sigma^*}, \rho^*, m_x^*)$ to $\mathcal{A}_1$. In the simulation, the challenge ciphertext implicitly sets $\omega_1^* = e(g, \hat{g})^{abcv^* \cdot H_4(\eta^*)}$, $\omega_2^* = e(g, \hat{g})^{abct^* \cdot H_4(\eta^*)}$, $H_6(\omega_1^*, \eta^*, C_0^*,$

$C_1^*, C_2^*) = (m_x \parallel k) \oplus C_3^*$, $H_8(\omega_2^*) = \frac{C_4^*}{H_7(m_x)^R}$, where $g_1 = g^a$, $H_2(\rho^*) = \hat{g}^{bv^*}$, $H_3(\rho^*) = \hat{g}^{bt^*}$, $H_1(\sigma^*) = g^{u^*}$, $ek_{\sigma^*} = g^{su^*}$, $\eta^* = e(g,\hat{g})^{bsu^*t^*}$, $C_0^* = g^R$, $C_1^* = g^c$, and $C_2^* = g^{\beta_0' c}$. Finally, in the guess phase, $\mathcal{A}_1$ outputs a guess $x' \in \{0,1\}$. The advantage of $\mathcal{A}_1$ for breaking our proposal is defined as $\epsilon = |\Pr[x = x'] - \frac{1}{2}|$. If ϵ is non-negligible, then the tuple $[\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*, \delta^*]$ is documented in L_{H_6} with non-negligible probability. If $\mathcal{B}$ selects the right tuple from L_{H_6}, $\mathcal{B}$ can return the BDH instance solution $\omega_1^{*(v^*H_4(\eta^*))^{-1}}$ $(= e(g,\hat{g})^{abc})$. As a result, the BDH assumption can be addressed by $\mathcal{B}$ with non-negligible advantage if $\mathcal{A}_1$ is able to break our proposal with non-negligible advantage.

Subsequently, as for the confidentiality, we used the BDH assumption to prove that our proposal meets OW-ID-CCA security regarding the Type-II adversary $\mathcal{A}_2$. Given a BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b)$, we generated a simulated scheme $\mathcal{B}$ and interacted with $\mathcal{A}_2$ by following the OW-ID-CCA security model defined in Section 3.3. $\mathcal{B}$ simulates the oracles $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, and $\mathcal{O}_{Dec}$ to answer $\mathcal{A}_2$'s queries and preserves the L_H and $L_{H_i}(i = 1,2,3,5,6,7,8)$ lists to simulate the random oracles $\mathcal{O}_H$ and $\mathcal{O}_{H_i}(i = 1,2,3,5,6,7,8)$. In the challenge phase, $\mathcal{A}_2$ sends identities σ^*, rcv^* to $\mathcal{B}$. Let $rcv^* = \rho^*$. $\mathcal{B}$ randomly chooses a message $m^* \in \{0,1\}^\lambda$ and answers the challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*) = Enc(pp, ek_{\sigma^*}, \rho^*, m^*)$ to $\mathcal{A}_2$. In the simulation, the challenge ciphertext implicitly sets $\omega_1^* = e(g,\hat{g})^{abcv^* \cdot H_4(\eta^*)}$, $H_6(\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*) = (m^* \parallel k) \oplus C_3^*$, where $g_1 = g^a$, $H_2(\rho^*) = \hat{g}^{bv^*}$, $H_3(\rho^*) = \hat{g}^{t^*}$, $H_1(\sigma^*) = g^{u^*}$, $ek_{\sigma^*} = g^{su^*}$, $\eta^* = e(g,\hat{g})^{bsu^*t^*}$, $C_0^* = g^R$, $C_1^* = g^c$, $C_2^* = g^{\beta_0' c}$, and $C_4^* = H_7(m^*)^R \cdot H_8(e(g^c, \hat{g}^{at^* \cdot H_4(\eta^*)}))$. Finally, in the guess phase, $\mathcal{A}_2$ outputs a guess m'. The advantage of $\mathcal{A}_2$ for breaking our proposal is defined as $\epsilon = |\Pr[m^* = m']|$. If ϵ is non-negligible, then the tuple $[\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*, \delta^*]$ is documented in L_{H_6} with non-negligible probability. If $\mathcal{B}$ selects the right tuple from L_{H_6}, $\mathcal{B}$ can return the BDH instance solution $\omega_1^{*(v^*H_4(\eta^*))^{-1}} (= e(g,\hat{g})^{abc})$. As a result, the BDH assumption can be addressed by $\mathcal{B}$ with non-negligible advantage if $\mathcal{A}_2$ is able to break our proposal with non-negligible advantage.

As for the anonymity, we used the Gap-BDH assumption to prove that our proposal meets ANON-ID-CCA security regarding the Type-III adversary $\mathcal{A}_3$. Given a Gap-BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \mathcal{O}_{\mathcal{DBDH}})$, we generated a simulated scheme $\mathcal{B}$ and interacted with $\mathcal{A}_3$ by following the ANON-ID-CCA security model defined in Section 3.3. $\mathcal{B}$ simulates the oracles $\mathcal{O}_H$, $\mathcal{O}_{H_i}(i = 1,2,3,4,5,6,7,8)$, $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Enc}$, and $\mathcal{O}_{Dec}$ to answer $\mathcal{A}_3$'s queries. In the challenge phase, $\mathcal{A}_3$ sends identities (snd_0^*, ρ_0^*), (snd_1^*, ρ_1^*) and a message m^* to $\mathcal{B}$. Let $snd_0^* = \sigma_0^*, snd_1^* = \sigma_1^*$. $\mathcal{B}$ randomly chooses $x \in \{0,1\}$ and answers the challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*) = Enc(pp, ek_{\sigma_x^*}, \rho_x^*, m^*)$ and the challenge trapdoor $td_{(\sigma_x^*, \rho_x^*)} = (y_1, y_2) = Auth(pp, \sigma_x^*, dk_{\rho_x^*})$ to $\mathcal{A}_3$. In the simulation, the challenge ciphertext implicitly sets $\eta^* = e(g,\hat{g})^{abcu_x^*t_x^*}$, $\omega_1^* = e(g^{a\alpha'}, \hat{g}^b)^{r\Omega v_x^*}$, $C_3^* = (m^* \parallel k) \oplus H_6(\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*)$, where $g_1 = g^{a\alpha'}$, $H_1(\sigma_0^*) = g^{cu_{i_x^*}}$, $H_2(\rho_x^*) = \hat{g}^{bv_{j_t^*}}$, $H_3(\rho_x^*) = \hat{g}^{bv_{t_x^*}}$, $\omega_2^* = e(g^{a\alpha'}, \hat{g}^b)^{r\tilde{\Omega}_{xx}}$, $H(\eta^*) = I = I_{xx}$, $H_4(\eta^*) = \Omega = \frac{\tilde{\Omega}_{xx}}{t_x^*}$, $C_0^* = g^R$, $C_1^* = g^r$, $C_2^* = (fh^I)^r$, and $C_4^* = H_7(m^*)^R \cdot H_8(\omega_2^*)$ Furthermore, the challenge trapdoor implicitly sets $y = \tilde{y} - bz$, where $z = \frac{t_x\alpha'\Omega}{\beta_1 I} = \frac{\alpha'\tilde{\Omega}_{xx}}{\beta_1 I}$, $y_1 = \hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}}$, $y_2 = \hat{g}^{\tilde{y}-bz}$. Finally, in the guess phase, $\mathcal{A}_3$ outputs a guess $x' \in \{0,1\}$. The advantage of $\mathcal{A}_3$ for breaking our proposal is defined as $\epsilon = |\Pr[x = x'] - \frac{1}{2}|$. If ϵ is non-negligible, $\eta^* = e(g,\hat{g})^{abcu_x^*t_x^*}$ has been queried to $\mathcal{O}_H$ with non-negligible probability. With $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^{*(u_{i_x^*}t_{j_x^*})^{-1}}) = 1$, $\mathcal{B}$ can return the Gap-BDH instance solution $\eta^{*(u_{i_x^*}t_{j_x^*})^{-1}}$ $(= e(g,\hat{g})^{abc})$. As a result, the Gap-BDH assumption can be addressed by $\mathcal{B}$ with non-negligible advantage if $\mathcal{A}_3$ is able to break our proposal with non-negligible advantage.

As for the authenticity, we used the Gap-BDH assumption to prove that our proposal meets sUF-ID-CMA security regarding the Type-IV adversary $\mathcal{A}_4$. Given a Gap-BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \mathcal{O}_{\mathcal{DBDH}})$, we generated a simulated scheme $\mathcal{B}$ and interacted with $\mathcal{A}_4$ by following the sUF-ID-CMA security model defined in Section 3.3. $\mathcal{B}$ simulates the oracles $\mathcal{O}_H$, $\mathcal{O}_{H_i}(i = 1,2,3,4,5,6,7,8)$, $\mathcal{O}_{SKGen}$, $\mathcal{O}_{RKGen}$, $\mathcal{O}_{Auth}$, $\mathcal{O}_{Enc}$, and

$\mathcal{O}_{Dec}$ to answer $\mathcal{A}_4$'s queries. In the simulation, the following numbers are implicitly set $\eta^* = e(g,\hat{g})^{abc}$, where $H_1(\sigma^*) = g^c$, $H_3(\rho^*) = \hat{g}^b$, $H(\eta^*) = I^*$, $H_4(\eta^*) = \Omega^*$. In the forgery phase, $\mathcal{A}_4$ outputs a triple (snd^*, ρ^*, C^*), where $snd^* = \sigma^*$ and $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$. If $m^* = Dec(pp, dk_{\rho^*}, \sigma^*, C^*) \neq \perp$, $\mathcal{A}_4$ wins. The advantage of $\mathcal{A}_4$ for breaking our proposal is defined as $\epsilon = \Pr[\mathcal{A}_4 \text{ wins}]$. With ϵ and the lemma on the relationship between the chosen-identity attack and given identity attack [33], if ϵ is non-negligible, $\eta^* = e(g,\hat{g})^{abc}$ has been queried to $\mathcal{O}_H$ with non-negligible probability. Then, $\mathcal{O}_{\mathcal{BDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^*) = 1$, $\mathcal{B}$ can return the Gap-BDH instance solution η^* $(= e(g,\hat{g})^{abc})$. As a result, the Gap-BDH assumption can be addressed by $\mathcal{B}$ with non-negligible advantage if $\mathcal{A}_4$ is able to break our proposal with non-negligible advantage.

Theorem 1. *For any $\mathcal{A}_1$, our IBME-ET scheme meets IND-ID-CCA security on the basis of the BDH assumption.*

More precisely, if $\mathcal{A}_1$ is able to break our proposal with the advantage ϵ, we can conceive of a PPT algorithm $\mathcal{B}$ to address the BDH assumption with the advantage $\epsilon' \geq \frac{1}{q_{H_6}}(\frac{\epsilon}{q_{H_1} q_{H_2}} - \frac{q_D}{2^{\lambda+l}} - \frac{q_{H_8}}{q})$, where $q_{H_i}(i = 1,2,6,8)$ and q_D denote the numbers of different queries to $\mathcal{O}_{H_i}(i = 1,2,6,8)$ and $\mathcal{O}_{Dec}$, respectively.

Proof. Given a BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b)$, the task of $\mathcal{B}$ is to calculate $e(g,\hat{g})^{abc}$ by interacting with $\mathcal{A}_1$ as below:

(1) *Setup*: $\mathcal{B}$ randomly selects $i^* \in \{1,2,\cdots,q_{H_1}\}$, $j^* \in \{1,2,\cdots,q_{H_2}\}$. $\mathcal{B}$ randomly chooses $I^*, s, \beta_0', \beta_1' \in \mathbb{Z}_q^*$, calculates $g_1 = g^a$, $f = g^{\beta_0' - a\beta_1' I^*}$, $h = g^{a\beta_1'}$, $\hat{f} = \hat{g}^{\beta_0' - a\beta_1' I^*}$, and $\hat{h} = \hat{g}^{a\beta_1'}$, sets $pp = (\mathcal{G}, g, \hat{g}, g_1, f, h, \hat{f}, \hat{h}, H, H_i(i = 1,2,3,4,5,6,7,8))$, and delivers this to $\mathcal{A}_1$ with pp. $\mathcal{B}$ implicitly sets $mk = (s, a)$, because $\mathcal{B}$ has no knowledge about a. $\mathcal{B}$ preserves the L_H and $L_{H_i}(i = 1,2,3,5,6,7,8)$ lists to simulate $\mathcal{O}_H$ and $\mathcal{O}_{H_i}(i = 1,2,3,5,6,7,8)$. Afterwards, $\mathcal{B}$ randomly selects $u^*, v^*, t^* \in \mathbb{Z}_q^*$.

(2) *Phase 1*: $\mathcal{B}$ answers $\mathcal{A}_1$'s queries.

- $\mathcal{O}_H(\eta)$: When $\eta \neq e(g,\hat{g})^{bsu^*t^*}$, $\mathcal{B}$ randomly selects $I \in \mathbb{Z}_q^*$, inserts a tuple $[\eta, I]$ into L_H, and answers I. Otherwise, $\mathcal{B}$ answers I^*.
- $\mathcal{O}_{H_1}(\sigma_i)$: Suppose σ_i as the i-th different query. When $i \neq i^*$, $\mathcal{B}$ randomly selects $u_i \in \mathbb{Z}_q^*$, inserts a tuple $[\sigma_i, u_i]$ into L_{H_1}, and returns g^{u_i}. Otherwise, $\mathcal{B}$ has $u_{i^*} = u^*$, inserts a tuple $[\sigma_{i^*}, u_{i^*}]$ into L_{H_1}, and returns $g^{u_{i^*}}$.
- $\mathcal{O}_{H_2}(\rho_j)$: Suppose ρ_j as the j-th different query. When $j \neq j^*$, $\mathcal{B}$ randomly selects $v_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, v_j]$ into L_{H_2}, and returns $\hat{g}^{v_j}$. Otherwise, $\mathcal{B}$ has $v_{j^*} = v^*$, inserts a tuple $[\rho_{j^*}, v_{j^*}]$ into L_{H_2}, and returns $\hat{g}^{bv_{j^*}}$.
- $\mathcal{O}_{H_3}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$. Subsequently, $\mathcal{B}$ searches the tuple $[\rho_j, v_j]$ in L_{H_2}. When $j \neq j^*$, $\mathcal{B}$ selects $t_j \in \mathbb{Z}_q^*$ randomly, inserts a tuple $[\rho_j, t_j]$ into L_{H_3}, and returns $\hat{g}^{t_j}$. Otherwise, $\mathcal{B}$ has $t_{j^*} = t^*$, inserts a tuple $[\rho_{j^*}, t_{j^*}]$ into L_{H_3}, and returns $\hat{g}^{bt_{j^*}}$.
- $\mathcal{O}_{H_5}(m,k)$: $\mathcal{B}$ randomly chooses $R \in \mathbb{Z}_q^*$, inserts a tuple $[m,k,R]$ into L_{H_5}, and answers R.
- $\mathcal{O}_{H_6}(\omega_1, \eta, C_0, C_1, C_2)$: $\mathcal{B}$ randomly chooses $\delta \in \{0,1\}^{\lambda+l}$, inserts a tuple $[\omega_1, \eta, C_0, C_1, C_2, \delta]$ into L_{H_6}, and answers δ.
- $\mathcal{O}_{H_7}(m)$: $\mathcal{B}$ randomly selects $h_7 \in \hat{\mathbb{G}}$, inserts a tuple $[m, h_7]$ into L_{H_7}, and returns h_7.
- $\mathcal{O}_{H_8}(\omega_2)$: $\mathcal{B}$ randomly selects $\pi \in \hat{\mathbb{G}}$, inserts a tuple $[\omega_2, \pi]$ into L_{H_8}, and returns π.
- $\mathcal{O}_{SKGen}(\sigma_i)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\sigma_i, u_i]$ in L_{H_1}. Next, $\mathcal{B}$ returns $ek_{\sigma_i} = g^{su_i}$.
- $\mathcal{O}_{RKGen}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$. There are a tuple $[\rho_j, v_j]$ in L_{H_2} and a tuple $[\rho_j, t_j]$ in L_{H_3}. When $j \neq j^*$, $\mathcal{B}$ returns $dk_{\rho_j} = (d_1, d_2, d_3) = (\hat{g}^{st_j}, \hat{g}^{av_j}, \hat{g}^{at_j})$. Otherwise, $\mathcal{B}$ is aborted by failure.

- $\mathcal{O}_{Dec}(\rho_j, snd, C)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$.
  - When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j} and returns the outcome of the algorithm $Dec(pp, dk_{\rho_j}, \sigma_i, C)$.
  - Otherwise, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i} and calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$. For each tuple $[\omega_1, \eta, C_0, C_1, C_2, \delta]$ in L_{H_6}, $\mathcal{B}$ calculates $m' \parallel k' = C_3 \oplus \delta$ and calculates $R' = H_5(m', k')$. If $C_0 = g^{R'}$ and there exists a tuple $[\omega_2, \pi]$ in L_{H_8} such that $C_4 = H_7(m')^{R'} \cdot \pi$ holds, it outputs m'. Once L_{H_8} has no such tuple, $\mathcal{B}$ outputs $\perp$.
- $\mathcal{O}_{Auth}(snd, \rho_j)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$. When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j}, returns $td_{(\sigma_i,\rho_j)} = Auth(pp, \sigma_i, dk_{\rho_j})$. Otherwise, $\mathcal{B}$ executes the following operations:
  - When $(i, j) = (i^*, j^*)$, $\mathcal{B}$ is aborted by failure.
  - Otherwise, L_{H_2} has a tuple $[\rho_{j^*}, v_{j^*}]$ and L_{H_3} has a tuple $[\rho_{j^*}, t_{j^*}]$, and $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i}, calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$, $I = H(\eta)$ and $\Omega = H_4(\eta)$, randomly selects $\tilde{y} \in \mathbb{Z}_q^*$, calculates $z = \frac{t_{j^*}\Omega}{\beta_1'(I-I^*)}$, implicitly sets $y = \tilde{y} - bz$, and returns $td_{(\sigma_i,\rho_{j^*})} = (y_1, y_2) = (\hat{g}^{\beta_0'(\tilde{y}-bz)}\hat{g}^{a\beta_1'(I-I^*)\tilde{y}}, \hat{g}^{\tilde{y}-bz})$. $td_{(\sigma_i,\rho_{j^*})} = (y_1, y_2)$ is a valid random trapdoor according to ρ_{j^*} and σ_i, where

    $y_1 = \hat{g}^{\beta_0'(\tilde{y}-bz)}\hat{g}^{a\beta_1'(I-I^*)\tilde{y}} = \hat{g}^{abt_{j^*}\cdot\Omega}\hat{g}^{\beta_0' y}\hat{g}^{a\beta_1'(I-I^*)y} = d_3^{\Omega}\hat{g}^{(\beta_0'-a\beta_1 I^* + a\beta_1' I)y} = d_3^{\Omega}(\hat{f}\hat{h}^I)^y$,
    $y_2 = \hat{g}^{\tilde{y}-bz} = \hat{g}^y$.

(3) *Challenge*: $\mathcal{A}_1$ offers equal-length messages $m_0^*, m_1^* \in \{0,1\}^\lambda$ along with the pair of sender/receiver identities (σ^*, rcv^*) to $\mathcal{B}$. Let $rcv^* = \rho^*$. Afterwards, $\mathcal{B}$ utilizes a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma^*)$ and $\mathcal{O}_{H_3}(\rho^*)$.
  - When the i^*-th tuple in L_{H_1} is $[\sigma^*, u^*]$ and the j^*-th tuple in L_{H_2} is $[\rho^*, v^*]$, $\mathcal{B}$ randomly selects $x \in \{0,1\}$, $C_3^* \in \{0,1\}^{\lambda+l}$, $C_4^* \in \hat{\mathbb{G}}$ and $k \in \{0,1\}^l$, calculates $ek_{\sigma^*} = g^{su^*}$, $\eta^* = e(g, \hat{g})^{bsu^*t^*}$, $R = H_5(m_x, k)$, $C_0^* = g^R$, $C_1^* = g^c$, and $C_2^* = g^{\beta_0' c}$, and then, sends the challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$ to $\mathcal{A}_1$.
    The above construction implicitly sets $\omega_1^* = e(g, \hat{g})^{abcv^*\cdot H_4(\eta^*)}$, $\omega_2^* = e(g, \hat{g})^{abct^*\cdot H_4(\eta^*)}$, $H_6(\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*) = (m_x \parallel k) \oplus C_3^*$, $H_8(\omega_2^*) = \frac{C_4^*}{H_7(m_x)^R}$, where $g^{u^*} = H_1(\sigma^*)$, $\hat{g}^{bv^*} = H_2(\rho^*)$, $\hat{g}^{bt^*} = H_3(\rho^*)$.
  - Otherwise, $\mathcal{B}$ is aborted by failure.

(4) *Phase 2*: $\mathcal{A}_1$ makes queries like in *Phase* 1.

(5) *Guess*: $\mathcal{A}_1$ answers a guess $x' \in \{0,1\}$. $\mathcal{B}$ randomly selects a tuple $[\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*, \delta^*]$ from L_{H_6} and returns the BDH instance solution $\omega_1^{*(v^*H_4(\eta^*))^{-1}} (= e(g, \hat{g})^{abc})$.

□

Analysis: It is obvious that the simulations of $\mathcal{O}_H$, $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_5}$, and $\mathcal{O}_{H_7}$ are perfect. Denote the query $\mathcal{O}_{H_6}(e(g, \hat{g})^{abcv^*\cdot H_4(\eta^*)}, \eta^*, C_0^*, C_1^*, C_2^*)$ as the event $AskH_6^*$. Denote the query $\mathcal{O}_{H_8}(e(g, \hat{g})^{abct^*\cdot H_4(\eta^*)})$ as the event $AskH_8^*$. Denote the failure of $\mathcal{B}$ to decrypt the legitimate ciphertext in $\mathcal{O}_{Dec}$ as the event $Derr$. Thus, $\Pr[Derr] \leq \frac{q_D}{2^{\lambda+l}}$. Let $rcv^* = \rho^*$. Suppose $AbortRK$ as the event in which $\mathcal{B}$ terminates upon the query $\mathcal{O}_{RKGen}(\rho^*)$ being issued, $AbortAuth$ as the event in which $\mathcal{B}$ terminates upon the query $\mathcal{O}_{Auth}(\sigma^*, \rho^*)$ being issued, and $AbortCh$ as the event in which $\mathcal{B}$ terminates in the challenge phase. Clearly, $\neg AbortCh$ implies $\neg AbortRK$ and $\neg AbortAuth$, because the queries $\mathcal{O}_{RKGen}(\rho^*)$ and $\mathcal{O}_{Auth}(\sigma^*, \rho^*)$ cannot be issued. We obtain $\Pr[\neg AbortCh] \geq \frac{1}{q_{H_1} q_{H_2}}$.

Define $E = (AskH_6^* \vee AskH_8^* \vee Derr) | \neg AbortCh$. There is no greater over $\frac{1}{2}$ advantage that $\mathcal{A}_1$ will gain in guessing x when E does not happen because $\mathcal{O}_{H_6}$ and $\mathcal{O}_{H_8}$ are random oracles. $\Pr[x = x' | \neg E] = \frac{1}{2}$. Hence,

$$\Pr[x = x'] = \Pr[x = x'|\neg E]\Pr[\neg E] + \Pr[x = x'|E]\Pr[E] \leq \frac{1}{2}\Pr[\neg E] + \Pr[E] = \frac{1}{2} + \frac{1}{2}\Pr[E].$$

With ϵ, we obtain

$$\epsilon = |\Pr[x = x'] - \frac{1}{2}| \leq \Pr[E] \leq \frac{\Pr[AskH_6^*] + \Pr[AskH_8^*] + \Pr[Derr]}{\Pr[\neg AbortCh]}.$$

Subsequently, we obtain

$$\Pr[AskH_6^*] \geq \epsilon\Pr[\neg AbortCh] - \Pr[Derr] - \Pr[AskH_8^*] \geq \frac{\epsilon}{q_{H_1}q_{H_2}} - \frac{q_D}{2^{\lambda+l}} - \frac{q_{H_8}}{q}.$$

When $AskH_6^*$ happens, $\mathcal{A}_1$ can distinguish the simulation of the challenge ciphertext C^*. Because $\mathcal{O}_{H_6}(e(g,\hat{g})^{abcv^* \cdot H_4(\eta^*)}, \eta^*, C_0^*, C_1^*, C_2^*)$ has been documented in L_{H_6} with non-negligible probability, $\mathcal{B}$ is winning when the right element is selected from L_{H_6}. Thus, the BDH assumption can be addressed by $\mathcal{B}$ with advantage $\epsilon' \geq \frac{1}{q_{H_6}}\Pr[AskH_6^*] \geq \frac{1}{q_{H_6}}(\frac{\epsilon}{q_{H_1}q_{H_2}} - \frac{q_D}{2^{\lambda+l}} - \frac{q_{H_8}}{q})$.

Theorem 2. *For any $\mathcal{A}_2$, our IBME-ET scheme meets OW-ID-CCA security on the basis of the BDH assumption.*

More precisely, if $\mathcal{A}_2$ is able to break our proposal with the advantage ϵ, we are able to conceive of a PPT algorithm $\mathcal{B}$ to address the BDH assumption with the advantage $\epsilon' \geq \frac{1}{q_{H_6}}(\frac{\epsilon - \frac{1}{2^\lambda}}{q_{H_1}q_{H_2}} - \frac{q_D}{2^{\lambda+l}})$, where $q_{H_i}(i = 1,2,6)$ and q_D denote the numbers of different queries to $\mathcal{O}_{H_i}(i = 1,2,6)$ and $\mathcal{O}_{Dec}$, respectively.

Proof. Given a BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b)$, the task of $\mathcal{B}$ is to calculate $e(g,\hat{g})^{abc}$ by interacting with $\mathcal{A}_2$ as below:

(1) *Setup*: $\mathcal{B}$ executes like in the proof of Theorem 1.
(2) *Phase 1*: $\mathcal{B}$ answers $\mathcal{A}_2$'s queries.

- For $\mathcal{O}_H(\eta)$, $\mathcal{O}_{H_1}(\sigma_i)$, $\mathcal{O}_{H_2}(\rho_j)$, $\mathcal{O}_{H_5}(m,k)$, $\mathcal{O}_{H_6}(\omega_1,\eta,C_0,C_1,C_2)$, $\mathcal{O}_{H_7}(m)$, and $\mathcal{O}_{H_8}(\omega_2)$, $\mathcal{B}$ executes like in the proof of Theorem 1.
- $\mathcal{O}_{H_3}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$. Subsequently, $\mathcal{B}$ searches the tuple $[\rho_j, v_j]$ in L_{H_2}. When $j \neq j^*$, $\mathcal{B}$ randomly selects $t_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, t_j]$ into L_{H_3}, and returns $\hat{g}^{t_j}$. Otherwise, $\mathcal{B}$ sets $t_{j^*} = t^*$, inserts a tuple $[\rho_{j^*}, t_{j^*}]$ into L_{H_3}, and returns $\hat{g}^{t_{j^*}}$.
- $\mathcal{O}_{SKGen}(\sigma_i)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\sigma_i, u_i]$ in L_{H_1}. Next, $\mathcal{B}$ returns $ek_{\sigma_i} = g^{su_i}$.
- $\mathcal{O}_{RKGen}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$. There are a tuple $[\rho_j, v_j]$ in L_{H_2} and a tuple $[\rho_j, t_j]$ in L_{H_3}. When $j \neq j^*$, $\mathcal{B}$ returns $dk_{\rho_j} = (d_1, d_2, d_3) = (\hat{g}^{st_j}, \hat{g}^{av_j}, \hat{g}^{at_j})$. Otherwise, $\mathcal{B}$ is aborted by failure.
- $\mathcal{O}_{Dec}(\rho_j, snd, C)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$.
  - When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j} and returns the outcome of the algorithm $Dec(pp, dk_{\rho_j}, \sigma_i, C)$.
  - Otherwise, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i} and calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$. For each tuple $[\omega_1, \eta, C_0, C_1, C_2, \delta]$ in L_{H_6}, $\mathcal{B}$ calculates $m' \parallel k' = C_3 \oplus \delta$ and calculates $R' = H_5(m', k')$. If $C_0 = g^{R'}$ and there exists a tuple $[\omega_2, \pi]$ in L_{H_8} such that $C_4 = H_7(m')^{R'} \cdot \pi$ holds, it outputs m'. When L_{H_8} has no such tuple, $\mathcal{B}$ outputs $\perp$.
- $\mathcal{O}_{Auth}(snd, \rho_j)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$.

- When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j} and returns $td_{(\sigma_i,\rho_j)} = Auth(pp, \sigma_i, dk_{\rho_j})$.
- Otherwise, there are a tuple $[\rho_{j^*}, v_{j^*}]$ in L_{H_2} and a tuple $[\rho_{j^*}, t_{j^*}]$ in L_{H_3}, and $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i}, calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$, $I = H(\eta)$, $\Omega = H_4(\eta)$, and $d_3 = H_3(\rho_{j^*})^a = \hat{g}^{av_{j^*}}$, randomly selects $y \in \mathbb{Z}_q^*$, and returns $td_{(\sigma_i,\rho_{j^*})} = (y_1, y_2) = (d_3^{\Omega}(\hat{f}\hat{h}^I)^y, \hat{g}^y)$.

(3) *Challenge*: $\mathcal{A}_2$ submits a pair of sender/receiver identities (σ^*, rcv^*) to $\mathcal{B}$. Let $rcv^* = \rho^*$. Afterwards, $\mathcal{B}$ chooses a message $m^* \in \{0,1\}^\lambda$ randomly and executes a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma^*)$ and $\mathcal{O}_{H_3}(\rho^*)$.

- When the i^*-th tuple in L_{H_1} is $[\sigma^*, u^*]$ and the j^*-th tuple in L_{H_2} is $[\rho^*, v^*]$, $\mathcal{B}$ randomly selects $k \in \{0,1\}^l$, $C_3^* \in \{0,1\}^{\lambda+l}$, calculates $ek_{\sigma^*} = g^{su^*}$, $\eta^* = e(g,\hat{g})^{bsu^*t^*}$, $R = H_5(m_x, k)$, $C_0^* = g^R$, $C_1^* = g^c$, $C_2^* = g^{\beta_0' c}$, and $C_4^* = H_7(m^*)^R \cdot H_8(e(g^c, \hat{g}^{at^* \cdot H_4(\eta^*)}))$, and delivers this to $\mathcal{A}_2$ with the challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$.
  The above construction implicitly sets $\omega_1^* = e(g,\hat{g})^{abcv^* \cdot H_4(\eta^*)}$, $H_6(\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*) = (m^* \parallel k) \oplus C_3^*$, where $g^{u^*} = H_1(\sigma^*)$, $\hat{g}^{bv^*} = H_2(\rho^*)$, $\hat{g}^{t^*} = H_3(\rho^*)$.
- Otherwise, $\mathcal{B}$ is aborted by failure.

(4) *Phase 2*: $\mathcal{A}_2$ makes issues like in *Phase 1*.

(5) *Guess*: $\mathcal{A}_2$ answers a guess m'. $\mathcal{B}$ randomly chooses a tuple $[\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*, \delta^*]$ from L_{H_6} and answers the BDH instance solution $\omega_1^{*(v^* H_4(\eta^*))^{-1}} (= e(g,\hat{g})^{abc})$.

□

Analysis: It is obvious that the simulations of $\mathcal{O}_H$, $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_5}$, $\mathcal{O}_{H_7}$, and $\mathcal{O}_{H_8}$ are perfect. Denote the query $\mathcal{O}_{H_6}(e(g,\hat{g})^{abcv^* \cdot H_4(\eta^*)}, \eta^*, C_0^*, C_1^*, C_2^*)$ as the event $AskH_6^*$. Denote the failure of $\mathcal{B}$ to decrypt the legitimate ciphertext in $\mathcal{O}_{Dec}$ as the event *Derr*. Hence, we have, $\Pr[Derr] \leq \frac{q_D}{2^{\lambda+l}}$. Let $rcv^* = \rho^*$. Suppose *AbortRK* as the event in which $\mathcal{B}$ terminates upon the query $\mathcal{O}_{RKGen}(\rho^*)$ being issued and *AbortCh* the event in which $\mathcal{B}$ terminates in the challenge phase. Clearly, $\neg AbortCh$ implies $\neg AbortRK$, because the query $\mathcal{O}_{RKGen}(\rho^*)$ cannot be issued. We obtain $\Pr[\neg AbortCh] \geq \frac{1}{q_{H_1} q_{H_2}}$.

Define $E = (AskH_6^* \vee Derr) | \neg AbortCh$. There is no greater over $\frac{1}{2^\lambda}$ advantage that $\mathcal{A}_2$ will gain in guessing m when E does not happen, because $\mathcal{O}_{H_6}$ is a random oracle. $\Pr[m = m' | \neg E] \leq \frac{1}{2^\lambda}$. Hence,

$$\begin{aligned}\Pr[m = m'] &= \Pr[m = m'|\neg E]\Pr[\neg E] + \Pr[m = m'|E]\Pr[E] \\ &\leq \frac{1}{2^\lambda}\Pr[\neg E] + \Pr[E] = \frac{1}{2^\lambda} + \frac{1}{2}\Pr[E] \\ &= (1 - \frac{1}{2^\lambda})\Pr[E] + \frac{1}{2^\lambda}.\end{aligned}$$

With ϵ, we obtain

$$\epsilon = |\Pr[m = m']| \leq (1 - \frac{1}{2^\lambda})\Pr[E] + \frac{1}{2^\lambda} \leq (1 - \frac{1}{2^\lambda})\frac{\Pr[AskH_6^*] + \Pr[Derr]}{\Pr[\neg AbortCh]} + \frac{1}{2^\lambda}.$$

Subsequently, we obtain

$$\Pr[AskH_6^*] \geq \frac{\epsilon - \frac{1}{2^\lambda}}{1 - \frac{1}{2^\lambda}} \Pr[\neg AbortCh] - \Pr[Derr] \geq \frac{\epsilon - \frac{1}{2^\lambda}}{q_{H_1} q_{H_2}} - \frac{q_D}{2^{\lambda+l}}.$$

When $AskH_6^*$ happens, $\mathcal{A}_2$ can distinguish the simulation of the challenge ciphertext C^*. Because $[e(g,\hat{g})^{abcv^* \cdot H_4(\eta^*)}, \eta^*, C_0^*, C_1^*, C_2^*, \delta^*]$ has been documented in L_{H_6} with non-negligible probability, $\mathcal{B}$ is winning when the right element is selected from L_{H_6}. Thus, the BDH assumption can be addressed by $\mathcal{B}$ with advantage $\epsilon' \geq \frac{1}{q_{H_6}} \Pr[AskH_6^*] \geq \frac{1}{q_{H_6}} (\frac{\epsilon - \frac{1}{2^\lambda}}{q_{H_1} q_{H_2}} - \frac{q_D}{2^{\lambda+l}})$.

Theorem 3. *For any $\mathcal{A}_3$, our IBME-ET scheme meets ANON-ID-CCA security on the basis of the Gap-BDH assumption.*

More precisely, if $\mathcal{A}_3$ is able to break our proposal with the advantage ϵ, we are able to conceive of a PPT algorithm $\mathcal{B}$ to address the Gap-BDH assumption with the advantage $\epsilon' \geq \frac{\epsilon}{q_{H_1}^2 q_{H_2}^2} - \frac{q_D}{2^{\lambda+l}}$, where $q_{H_i} (i = 1, 2)$ and q_D denote the numbers of different queries to $\mathcal{O}_{H_i} (i = 1, 2)$ and $\mathcal{O}_{Dec}$, respectively.

Proof. Given a Gap-BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \mathcal{O}_{\mathcal{DBDH}})$, the task of $\mathcal{B}$ is to calculate $e(g,\hat{g})^{abc}$ by interacting with $\mathcal{A}_3$ as below:

(1) *Setup*: $\mathcal{B}$ randomly selects $i_0^*, i_1^* \in \{1, 2, \cdots, q_{H_1}\}$ and $j_0^*, j_1^* \in \{1, 2, \cdots, q_{H_2}\}$. $\mathcal{B}$ randomly selects $\alpha', \beta_0, \beta_1 \in \mathbb{Z}_q^*$, calculates $g_1 = g^{a\alpha'}$, $f = g^{\beta_0}$, $h = g^{a\beta_1}$, $\hat{f} = \hat{g}^{\beta_0}$ and $\hat{h} = \hat{g}^{a\beta_1}$, sets $pp = (\mathcal{G}, g, \hat{g}, g_1, f, h, \hat{f}, \hat{h}, H, H_i (i = 1, 2, 3, 4, 5, 6, 7, 8))$, and delivers this to $\mathcal{A}_3$ with pp. $\mathcal{B}$ implicitly sets $mk = (s, \alpha) = (a, a\alpha')$, because $\mathcal{B}$ has no knowledge about a. $\mathcal{B}$ preserves the L_H, $L_{H_i} (i = 1, 2, 3, 4, 5, 6, 7, 8)$, and L_A lists to simulate $\mathcal{O}_H$, $\mathcal{O}_{H_i} (i = 1, 2, 3, 4, 5, 6, 7, 8)$, and $\mathcal{O}_{Auth}$. Afterwards, $\mathcal{B}$ randomly selects $u_{i_0^*}, u_{i_1^*}, v_{j_0^*}, v_{j_1^*}, t_{j_0^*}, t_{j_1^*} \in \mathbb{Z}_q^*$ and randomly chooses $\tilde{\Omega}_{00}, \tilde{\Omega}_{01}, \tilde{\Omega}_{11}, \tilde{\Omega}_{10}, I_{00}, I_{01}, I_{11}, I_{10} \in \mathbb{Z}_q^*$.

(2) *Phase 1*: $\mathcal{B}$ answers $\mathcal{A}_3$'s queries.

- $\mathcal{O}_H(\eta)$: $\mathcal{B}$ executes the following operations.
  - When $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^{(u_{i_0^*} t_{j_0^*})^{-1}}) = 1$, $\mathcal{B}$ returns the Gap-BDH instance solution $\eta^{(u_{i_0^*} t_{j_0^*})^{-1}}$ $(= e(g,\hat{g})^{abc})$ and defines $\Omega = \frac{\tilde{\Omega}_{00}}{t_{j_0^*}}$ and $I = I_{00}$.
  - When $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^{(u_{i_1^*} t_{j_1^*})^{-1}}) = 1$, $\mathcal{B}$ returns the Gap-BDH instance solution $\eta^{(u_{i_1^*} t_{j_1^*})^{-1}}$ $(= e(g,\hat{g})^{abc})$ and defines $\Omega = \frac{\tilde{\Omega}_{11}}{t_{j_1^*}}$ and $I = I_{11}$.
  - When $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^{(u_{i_0^*} t_{j_1^*})^{-1}}) = 1$, $\mathcal{B}$ returns the Gap-BDH instance solution $\eta^{(u_{i_0^*} t_{j_1^*})^{-1}}$ $(= e(g,\hat{g})^{abc})$ and defines $\Omega = \frac{\tilde{\Omega}_{01}}{t_{j_1^*}}$ and $I = I_{01}$.
  - When $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta^{(u_{i_1^*} t_{j_0^*})^{-1}}) = 1$, $\mathcal{B}$ returns the Gap-BDH instance solution $\eta^{(u_{i_1^*} t_{j_0^*})^{-1}}$ $(= e(g,\hat{g})^{abc})$ and defines $\Omega = \frac{\tilde{\Omega}_{10}}{t_{j_0^*}}$ and $I = I_{10}$.
  - Otherwise, $\mathcal{B}$ randomly selects $I, \Omega \in \mathbb{Z}_q^*$.

  Subsequently, $\mathcal{B}$ inserts $[\eta, \Omega]$ into L_{H_4} and $[\eta, I]$ into L_H and answers I.
- $\mathcal{O}_{H_1}(\sigma_i)$: Suppose σ_i as the i-th different query. When $i = i_0^*$, $\mathcal{B}$ inserts a tuple $[\sigma_{i_0^*}, u_{i_0^*}]$ into L_{H_1} and returns $g^{cu_{i_0^*}}$. When $i = i_1^*$, $\mathcal{B}$ inserts a tuple $[\sigma_{i_1^*}, u_{i_1^*}]$ into L_{H_1} and returns $g^{cu_{i_1^*}}$. Otherwise, $\mathcal{B}$ randomly selects $u_i \in \mathbb{Z}_q^*$, inserts a tuple $[\sigma_i, u_i]$ into L_{H_1}, and returns g^{u_i}.
- $\mathcal{O}_{H_2}(\rho_j)$: Suppose ρ_j as the j-th different query. When $j = j_0^*$, $\mathcal{B}$ inserts a tuple $[\rho_{j_0^*}, v_{j_0^*}]$ into L_{H_2} and returns $\hat{g}^{bv_{j_0^*}}$. When $j = j_1^*$, $\mathcal{B}$ inserts a tuple $[\rho_{j_1^*}, v_{j_1^*}]$ into

L_{H_2} and returns $\hat{g}^{bv_{j_1^*}}$. Otherwise, $\mathcal{B}$ randomly selects $v_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, v_j]$ into L_{H_2}, and returns $\hat{g}^{v_j}$.

- $\mathcal{O}_{H_3}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$. Subsequently, $\mathcal{B}$ searches the tuple $[\rho_j, v_j]$ in L_{H_2}. When $j = j_0^*$, $\mathcal{B}$ inserts a tuple $[\rho_{j_0^*}, t_{j_0^*}]$ into L_{H_3} and returns $\hat{g}^{bt_{j_0^*}}$. When $j = j_1^*$, $\mathcal{B}$ inserts a tuple $[\rho_{j_1^*}, t_{j_1^*}]$ into L_{H_3} and returns $\hat{g}^{bt_{j_1^*}}$. Otherwise, $\mathcal{B}$ randomly selects $t_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, t_j]$ into L_{H_3}, and returns $\hat{g}^{t_j}$.
- $\mathcal{O}_{H_4}(\eta)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_H(\eta)$. Subsequently, $\mathcal{B}$ searches for the tuple $[\eta, \Omega]$ in L_{H_4} and returns Ω.
- $\mathcal{O}_{H_5}(m, k)$: $\mathcal{B}$ randomly chooses $R \in \mathbb{Z}_q^*$, inserts a tuple $[m, k, R]$ into L_{H_5}, and answers R.
- $\mathcal{O}_{H_6}(\omega_1, \eta, C_0, C_1, C_2)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_H(\eta)$. Subsequently, $\mathcal{B}$ randomly selects $\delta \in \{0,1\}^{\lambda+l}$, inserts a tuple $[\omega_1, -, \eta, C_0, C_1, C_2, \delta]$ into L_{H_6}, and returns δ.
- $\mathcal{O}_{H_7}(m)$: $\mathcal{B}$ randomly selects $h_7 \in \hat{\mathbb{G}}$, inserts a tuple $[m, h_7]$ into L_{H_7}, and returns h_7.
- $\mathcal{O}_{H_8}(\omega_2)$: $\mathcal{B}$ randomly selects $\pi \in \hat{\mathbb{G}}$, inserts a tuple $[\omega_2, \pi]$ into L_{H_8}, and returns π.
- $\mathcal{O}_{SKGen}(\sigma_i)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\sigma_i, u_i]$ in L_{H_1}. When $i \neq i_0^*$ and $i \neq i_1^*$, $\mathcal{B}$ answers $ek_{\sigma_i} = g^{au_i}$. Otherwise, $\mathcal{B}$ is aborted by failure.
- $\mathcal{O}_{RKGen}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$. There are a tuple $[\rho_j, v_j]$ in L_{H_2} and a tuple $[\rho_j, t_j]$ in L_{H_3}. When $j \neq j_0^*$ and $j \neq j_1^*$, $\mathcal{B}$ answers $dk_{\rho_j} = (d_1, d_2, d_3) = (\hat{g}^{at_j}, \hat{g}^{a\alpha' v_j}, \hat{g}^{a\alpha' t_j})$. Otherwise, $\mathcal{B}$ is aborted by failure.
- $\mathcal{O}_{Enc}(\sigma_i, rcv, m)$: Let $rcv = \rho_j$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$ and $\mathcal{O}_{H_3}(\rho_j)$. When $i \neq i_0^*$ and $i \neq i_1^*$, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i} and returns $C = Enc(pp, ek_{\sigma_i}, \rho_j, m)$. Otherwise, $\mathcal{B}$ executes as below:
  - When $(i, j) = (i_0^*, j_0^*)$ or $(i, j) = (i_1^*, j_1^*)$, $\mathcal{B}$ is aborted by failure.
  - When $(i, j) = (i_0^*, j_1^*)$ or $(i, j) = (i_1^*, j_0^*)$, $\mathcal{B}$ executes a simulation algorithm to query $\mathcal{O}_{Auth}(\sigma_i, \rho_j)$. There is a tuple $[\sigma_i, \rho_j, I, \Omega, td_{(\sigma_i, \rho_j)}]$ in L_A. Afterwards, $\mathcal{B}$ randomly selects $r \in \mathbb{Z}_q^*$, $\delta \in \{0,1\}^{\lambda+l}$, $k \in \{0,1\}^l$, calculates $\omega_1 = e(g_1, H_2(\rho_j))^{r \cdot \Omega}$, $\omega_2 = e(g_1, H_3(\rho_j))^{r \cdot \Omega}$ and $R = H_5(m, k)$, inserts a tuple $[\omega_1, (i, j), -, C_0, C_1, C_2, \delta]$ into L_{H_6}, and returns $C = (C_0, C_1, C_2, C_3, C_4)$, where $C_0 = g^R$, $C_1 = g^r$, $C_2 = (fh^I)^r$, $C_3 = (m \parallel k) \oplus \delta$, $C_4 = H_7(m)^R \cdot H_8(\omega_2)$.
  - Otherwise, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to get $dk_{\rho_j} = (d_1, d_2, d_3)$, selects $k \in \{0,1\}^l$, $r \in \mathbb{Z}_q^*$ randomly, calculates $\eta = e(H_1(\sigma_i), d_1)$, $\omega_1 = e(g_1, H_2(\rho_j))^{r \cdot H_4(\eta)}$, $\omega_2 = e(g_1, H_3(\rho_j))^{r \cdot H_4(\eta)}$ and $R = H_5(m, k)$, and returns $C = (C_0, C_1, C_2, C_3, C_4)$, where $C_0 = g^R$, $C_1 = g^r$, $C_2 = (fh^{H(\eta)})^r$, $C_3 = (m \parallel k) \oplus H_6(\omega_1, \eta, C_0, C_1, C_2)$, $C_4 = H_7(m)^R \cdot H_8(\omega_2)$.
- $\mathcal{O}_{Dec}(\rho_j, snd, C)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$. When $j \neq j_0^*$ and $j \neq j_1^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j} and returns the outcome of $Dec(pp, dk_{(\rho_j)}, \sigma_i, C)$. Otherwise, $\mathcal{B}$ executes the following operations:
  - When $(i, j) = (i_0^*, j_0^*)$, or $(i, j) = (i_1^*, j_1^*)$m or $(i, j) = (i_0^*, j_1^*)$, or $(i, j) = (i_1^*, j_0^*)$, $\mathcal{B}$ searches for the tuple $[\sigma_i, \rho_j, I, \Omega, td_{(\sigma_i, \rho_j)}]$ in L_A. When L_A has no such tuple, $\mathcal{B}$ executes as below.
    When $(i, j) = (i_0^*, j_0^*)$, $\Omega = \frac{\tilde{\Omega}_{00}}{t_j}$, $I = I_{00}$. When $(i, j) = (i_1^*, j_1^*)$, $\Omega = \frac{\tilde{\Omega}_{11}}{t_j}$, $I = I_{11}$. When $(i, j) = (i_0^*, j_1^*)$, $\Omega = \frac{\tilde{\Omega}_{01}}{t_j}$, $I = I_{01}$. When $(i, j) = (i_1^*, j_0^*)$, $\Omega = \frac{\tilde{\Omega}_{10}}{t_j}$, $I = I_{10}$. Afterwards, $\mathcal{B}$ randomly selects $\tilde{y} \in \mathbb{Z}_q^*$, calculates $z = \frac{t_j \alpha' \Omega}{\beta_1 I}$, implicitly sets $y = \tilde{y} - bz$, sets $td_{(\sigma_i, \rho_j)} = (y_1, y_2) = (\hat{g}^{\beta_0(\tilde{y} - bz)} \hat{g}^{a\beta_1 I \tilde{y}}, \hat{g}^{\tilde{y} - bz})$,

and stores $[\rho_j, \sigma_i, I, \Omega, td_{(\sigma_i,\rho_j)}]$ in L_A. $td_{(\sigma_i,\rho_j)} = (y_1, y_2)$ is a valid random trapdoor according to ρ_j and σ_i, where

$$y_1 = \hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}} = \hat{g}^{a\alpha' bt_j\cdot\Omega}\hat{g}^{\beta_0 y}\hat{g}^{a\beta_1 Iy} = d_3^{\Omega}\hat{g}^{(\beta_0+a\beta_1 I)y} = d_3^{\Omega}(\hat{f}\hat{h}^I)^y,$$
$$y_2 = \hat{g}^{\tilde{y}-bz} = \hat{g}^y.$$

Next, $\mathcal{B}$ calculates $\omega_2 = \frac{e(C_1,y_1)}{e(C_2,y_2)}$. For each tuple $[\omega_1, (i,j), -, C_0, C_1, C_2, \delta]$ in L_{H_6}, $\mathcal{B}$ calculates $m \parallel k = C_3 \oplus \delta$ and $R = H_5(m,k)$. If both $C_0 = g^R$ and $C_4 = H_7(m)^R \cdot H_8(\omega_2)$ hold, $\mathcal{B}$ returns m; otherwise, $\mathcal{B}$ returns $\perp$.

- Otherwise, $\mathcal{B}$ can query $\mathcal{O}_{Auth}(\sigma_i, \rho_j)$ to obtain $td_{(\sigma_i,\rho_j)} = (y_1, y_2)$ and calculates $\omega_2 = \frac{e(C_1,y_1)}{e(C_2,y_2)}$. For each tuple $[\omega_1, (i,j), -, C_0, C_1, C_2, \delta]$ in L_{H_6}, $\mathcal{B}$ calculates $m \parallel k = C_3 \oplus \delta$ and $R = H_5(m,k)$. If both $C_0 = g^R$ and $C_4 = H_7(m)^R \cdot H_8(\omega_2)$ hold, $\mathcal{B}$ returns m; otherwise, $\mathcal{B}$ returns $\perp$.

- $\mathcal{O}_{Auth}(snd, \rho_j)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\rho_j, v_j]$ in L_{H_2}. When $j \neq j_0^*$ and $j \neq j_1^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain $dk_{\rho_j} = (d_1, d_2, d_3)$, calculates $\eta = e(H_1(\sigma_i), d_1)$, $I = H(\eta)$, $\Omega = H_3(\eta)$, returns $td_{(\sigma_i,\rho_j)} = Auth(pp, \sigma_i, dk_{\rho_j})$, and stores $[\sigma_i, \rho_j, I, \Omega, td_{(\sigma_i,\rho_j)}]$ into L_A. Otherwise, $\mathcal{B}$ executes as below:
  - When $(i,j) = (i_0^*, j_0^*)$ or $(i,j) = (i_1^*, j_1^*)$, $\mathcal{B}$ is aborted by failure.
  - When $(i,j) = (i_0^*, j_1^*)$, $\Omega = \frac{\tilde{\Omega}_{01}}{t_j}$, $I = I_{01}$.
  - When $(i,j) = (i_1^*, j_0^*)$, $\Omega = \frac{\tilde{\Omega}_{10}}{t_j}$, $I = I_{10}$.
  - Otherwise, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i} and calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$, $\Omega = H_4(\eta)$. $I = H(\eta)$

  Subsequently, $\mathcal{B}$ randomly selects $\tilde{y} \in \mathbb{Z}_q^*$, calculates $z = \frac{t_j\alpha'\Omega}{I\beta_1}$, implicitly sets $y = \tilde{y} - bz$, returns $td_{(\sigma_i,\rho_j)} = (y_1, y_2) = (\hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}}, \hat{g}^{\tilde{y}-bz})$, and then, stores $[\sigma_i, \rho_j, I, \Omega, td_{(\sigma_i,\rho_j)}]$ in L_A. $td_{(\sigma_i,\rho_j)} = (y_1, y_2)$ is a valid random trapdoor according to ρ_j and σ_i, where

$$y_1 = \hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}} = \hat{g}^{a\alpha' bt_j\cdot\Omega}\hat{g}^{\beta_0 y}\hat{g}^{a\beta_1 Iy} = d_3^{\Omega}\hat{g}^{(\beta_0+a\beta_1 I)y} = d_3^{\Omega}(\hat{f}\hat{h}^I)^y,$$
$$y_2 = \hat{g}^{\tilde{y}-bz} = \hat{g}^y.$$

(3) *Challenge*: $\mathcal{A}_3$ offers a message $m^* \in \{0,1\}^\lambda$ and two pairs of sender/receiver identities (snd_0^*, ρ_0^*), (snd_1^*, ρ_1^*) to $\mathcal{B}$. Set $snd_0^* = \sigma_0^*$, $snd_1^* = \sigma_1^*$. Afterwards, $\mathcal{B}$ utilizes a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_0^*)$, $\mathcal{O}_{H_1}(\sigma_1^*)$, $\mathcal{O}_{H_3}(\rho_0^*)$, and $\mathcal{O}_{H_3}(\rho_1^*)$:

- When the i_0^*-th tuple in L_{H_1} is $[\sigma_0^*, u_0^*]$, the i_1^*-th tuple in L_{H_1} is $[\sigma_1^*, u_1^*]$, the j_0^*-th tuple in L_{H_2} is $[\rho_0^*, v_0^*]$, and the j_1^*-th tuple in L_{H_2} is $[\rho_1^*, v_1^*]$, $\mathcal{B}$ executes the following operations:
  Firstly, $\mathcal{B}$ randomly selects $x \in \{0,1\}$ and searches for the tuple $[\sigma_x^*, \rho_x^*, I, \Omega, td_{(\sigma_x^*,\rho_x^*)}]$ in L_A. When L_A has no such tuple, $\mathcal{B}$ sets $\Omega = \frac{\tilde{\Omega}_{xx}}{t_x^*}$ and $I = I_{xx}$. Subsequently, $\mathcal{B}$ randomly selects $\tilde{y} \in \mathbb{Z}_q^*$, calculates $z = \frac{t_x\alpha'\Omega}{\beta_1 I} = \frac{\alpha'\tilde{\Omega}_{xx}}{\beta_1 I}$, implicitly sets $y = \tilde{y} - bz$, obtains $td_{(\sigma_x^*,\rho_x^*)} = (y_1, y_2) = (\hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}}, \hat{g}^{\tilde{y}-bz})$, and then, inserts a tuple $[\sigma_x^*, \rho_x^*, I, \Omega, td_{(\sigma_x^*,\rho_x^*)}]$ in L_A. $td_{(\sigma_x^*,\rho_x^*)} = (y_1, y_2)$ is a valid random trapdoor according to σ_x^* and ρ_x^*, where

$$y_1 = \hat{g}^{\beta_0(\tilde{y}-bz)}\hat{g}^{a\beta_1 I\tilde{y}} = \hat{g}^{ab\alpha'\cdot\tilde{\Omega}_{xx}}\hat{g}^{\beta_0 y}\hat{g}^{a\beta_1 Iy} = \hat{g}^{a\alpha' bt_x\cdot\Omega}\hat{g}^{\beta_0 y}\hat{g}^{a\beta_1 Iy} = d_3^{\Omega}(\hat{f}\hat{h}^I)^y,$$
$$y_2 = \hat{g}^{\tilde{y}-bz} = \hat{g}^y.$$

Secondly, $\mathcal{B}$ randomly selects $r \in \mathbb{Z}_q^*$, $C_3^* \in \{0,1\}^{\lambda+l}$, $k \in \{0,1\}^l$, calculates $\omega_2^* = e(g^{a\alpha'}, \hat{g}^b)^{r\tilde{\Omega}_{xx}}$, $R = H_5(m^*, k)$, $C_0^* = g^R$, $C_1^* = g^r$, $C_2^* = (fh^I)^r$, and $C_4^* = H_7(m^*)^R \cdot H_8(\omega_2^*)$.
The above construction implicitly sets $C_3^* = (m^* \parallel k) \oplus H_6(\omega_1^*, \eta^*, C_0^*, C_1^*, C_2^*)$, where $\omega_1^* = e(g^{a\alpha'}, \hat{g}^b)^{r\Omega v_x^*}$, $\eta^* = e(g, \hat{g})^{abcu_x^* t_x^*}$. $C_x^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$ is the encryption of m^* according to σ_x^* and ρ_x^*, where

$$\omega_2^* = e(g^{a\alpha'}, \hat{g}^b)^{r\tilde{\Omega}_{xx}} = e(g_1, \hat{g}^b)^{rt_x^*\Omega} = e(g_1, \hat{g}^{bt_x^*)^{r\cdot\Omega}} = e(g_1, H_3(\rho_x^*))^{r\cdot\Omega}.$$

Eventually, $\mathcal{B}$ returns the corresponding challenge ciphertext $C_x^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$ and challenge trapdoor $td_{(\sigma_x^*, \rho_x^*)} = (y_1, y_2)$ to $\mathcal{A}_3$.

- Otherwise, $\mathcal{B}$ is aborted by failure.

(4) *Phase 2*: $\mathcal{A}_3$ makes issues like in *Phase 1*.
(5) *Guess*: $\mathcal{A}_3$ answers a guess $x' \in \{0,1\}$.

□

Analysis: It is obvious that the simulations of $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_5}$, $\mathcal{O}_{H_7}$, and $\mathcal{O}_{H_8}$ are perfect. Define $\eta_{(0,0)} = e(g, \hat{g})^{abcu_{i_0^*} t_{j_0^*}}$, $\eta_{(1,1)} = e(g, \hat{g})^{abcu_{i_1^*} t_{j_0^*}}$, $\eta_{(1,0)} = e(g, \hat{g})^{abcu_{i_1^*} t_{j_0^*}}$, $\eta_{(0,1)} = e(g, \hat{g})^{abcu_{i_0^*} t_{j_1^*}}$. Let $snd_0^* = \sigma_0^*$ and $snd_1^* = \sigma_1^*$. Denote the queries $\mathcal{O}_H(\eta_{(0,0)})$, $\mathcal{O}_H(\eta_{(0,1)})$, $\mathcal{O}_H(\eta_{(1,0)})$, and $\mathcal{O}_H(\eta_{(1,1)})$ as the event *AskH*. Suppose *AbortSK* as the event in which $\mathcal{B}$ terminates upon the queries $\mathcal{O}_{SKGen}(\sigma_0^*)$ and $\mathcal{O}_{SKGen}(\sigma_1^*)$ being issued, *AbortRK* as the event in which $\mathcal{B}$ terminates upon the queries $\mathcal{O}_{RKGen}(\rho_0^*)$ and $\mathcal{O}_{RKGen}(\rho_1^*)$ being issued, *AbortAuth* as the event in which $\mathcal{B}$ terminates upon the queries $\mathcal{O}_{Auth}(\sigma_0^*, \rho_0^*)$ and $\mathcal{O}_{Auth}(\sigma_1^*, \rho_1^*)$ being issued, *AbortEnc* as the event in which $\mathcal{B}$ terminates upon the queries $\mathcal{O}_{Enc}(\sigma_0^*, \rho_0^*, *)$ and $\mathcal{O}_{Enc}(\sigma_1^*, \rho_1^*, *)$ being issued, and *AbortCh* as the event in which $\mathcal{B}$ terminates in the challenge phase. Clearly, $\neg AbortCh$ implies $\neg AbortSK$, $\neg AbortRK$, $\neg AbortAuth$, and $\neg AbortEnc$, because the queries $\mathcal{O}_{SKGen}(\sigma_0^*)$ and $\mathcal{O}_{SKGen}(\sigma_1^*)$ cannot be issued, the queries $\mathcal{O}_{RKGen}(\rho_0^*)$ and $\mathcal{O}_{RKGen}(\rho_1^*)$ are unable to be issued, (σ_0^*, ρ_0^*) and the queries $\mathcal{O}_{Auth}(\sigma_0^*, \rho_0^*)$ and $\mathcal{O}_{Auth}(\sigma_1^*, \rho_1^*)$ are unable to be issued, and the queries $\mathcal{O}_{Enc}(\sigma_0^*, \rho_0^*, *)$ and $\mathcal{O}_{Enc}(\sigma_1^*, \rho_1^*, *)$ are unable to be issued. Thus, we obtain $\Pr[\neg AbortCh] \geq \frac{1}{q_{H_1}^2} \cdot \frac{1}{q_{H_2}^2}$.

Denote the failure of $\mathcal{B}$ to decrypt the legitimate ciphertext in $\mathcal{O}_{Dec}$ as the event *Derr*. Thus, $\Pr[Derr] \leq \frac{q_D}{2^{\lambda+l}}$.

Define $E_0 = (AskH \vee Derr) | \neg AbortCh$. There is no greater over $\frac{1}{2}$ advantage that $\mathcal{A}_3$ will gain in guessing x when E_0 does not happen because $\mathcal{O}_H$, $\mathcal{O}_{H_4}$, and $\mathcal{O}_{H_6}$ are random oracles. Hence, $\Pr[x = x' | \neg E_0] = \frac{1}{2}$. We obtain

$$\Pr[x = x'] = \Pr[x = x' | \neg E_0]\Pr[\neg E_0] + \Pr[x = x' | E_0]\Pr[E_0] \leq \frac{1}{2}\Pr[\neg E_0] + \Pr[E_0] = \frac{1}{2} + \frac{1}{2}\Pr[E_0].$$

With ϵ, we obtain

$$\epsilon = |\Pr[x = x'] - \frac{1}{2}| \leq \Pr[E_0] \leq \frac{\Pr[AskH] + \Pr[Derr]}{\Pr[\neg AbortCh]}.$$

Subsequently, we obtain

$$\Pr[AskH] \geq \epsilon\Pr[\neg AbortCh] - \Pr[Derr] \geq \frac{\epsilon}{q_{H_1}^2 q_{H_2}^2} - \frac{q_D}{2^{\lambda+l}}.$$

Obviously, when *AskH* occurs, the Gap-BDH assumption can certainly be addressed by $\mathcal{B}$. $\mathcal{B}$ addresses the Gap-BDH assumption with advantage $\epsilon' = \Pr[\mathcal{B}\ success] = \Pr[AskH] \geq \frac{\epsilon}{q_{H_1}^2 q_{H_2}^2} - \frac{q_D}{2^{\lambda+l}}$.

Theorem 4. *For any $\mathcal{A}_4$, our IBME-ET scheme meets sUF-ID-CMA security on the basis of the Gap-BDH assumption.*

More precisely, if $\mathcal{A}_4$ is able to break our proposal with the advantage ϵ, we are able to conceive of a PPT algorithm $\mathcal{B}$ to address the Gap-BDH assumption with the advantage $\epsilon' \geq \epsilon(1-\frac{1}{q})\frac{1}{q_{H_1}q_{H_3}} - \frac{1+q_D}{2^\lambda}$, where $q_{H_i}(i=1,3)$ and q_D denote the numbers of different queries to $\mathcal{O}_{H_i}(i=1,3)$ and $\mathcal{O}_{Dec}$, respectively.

Proof. Given a Gap-BDH assumption instance $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \mathcal{O}_{\mathcal{DBDH}})$, the task of $\mathcal{B}$ is to calculate $e(g,\hat{g})^{abc}$ by interacting with $\mathcal{A}_4$ as below:

(1) *Setup*: $\mathcal{B}$ randomly chooses $i^* \in \{1,2,\cdots,q_{H_1}\}$, $j^* \in \{1,2,\cdots,q_{H_3}\}$. $\mathcal{B}$ randomly selects $\alpha, \beta_0, \beta_1 \in \mathbb{Z}_q^*$, calculates $g_1 = g^\alpha$, $f = g^{\beta_0}$, $h = g^{\beta_1}$, $\hat{f} = \hat{g}^{\beta_0}$ and $\hat{h} = \hat{g}^{\beta_1}$, sets $pp = (\mathcal{G}, g, \hat{g}, g_1, f, h, \hat{f}, \hat{h}, H, H_i(i=1,2,3,4,5,6,7,8))$, and delivers this to $\mathcal{A}_4$ with pp. $\mathcal{B}$ implicitly sets $mk = (a, \alpha)$, because $\mathcal{B}$ has no knowledge about a. $\mathcal{B}$ preserves the L_H and $L_{H_i}(i=1,2,3,4,5,6,7,8)$ lists to simulate $\mathcal{O}_H$ and $\mathcal{O}_{H_i}(i=1,2,3,4,5,6,7,8)$. Afterwards, $\mathcal{B}$ randomly selects $I^*, \Omega^* \in \mathbb{Z}_q^*$.

(2) *Queries:* $\mathcal{B}$ answers $\mathcal{A}_4$'s queries as below:

- $\mathcal{O}_H(\eta)$: When $\mathcal{O}_{\mathcal{DBDH}}(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \eta) \neq 1$, $\mathcal{B}$ randomly selects $\Omega, I \in \mathbb{Z}_q^*$, inserts $[\eta, \Omega]$ into L_{H_4} and $[\eta, I]$ into L_H, and answers I. Otherwise, $\mathcal{B}$ answers the Gap-BDH solution η $(= e(g,\hat{g})^{abc})$, defines $\Omega = \Omega^*$ and $I = I^*$, inserts $[\eta, \Omega]$ into L_{H_4} and $[\eta, I]$ into L_H, and answers I.
- $\mathcal{O}_{H_1}(\sigma_i)$: Suppose σ_i as the i-th different query. When $i \neq i^*$, $\mathcal{B}$ randomly selects $u_i \in \mathbb{Z}_q^*$, inserts a tuple $[\sigma_i, u_i]$ into L_{H_1}, returns g^{u_i}. Otherwise, $\mathcal{B}$ inserts a tuple $[\sigma_i, -]$ into L_{H_1} and returns g^c.
- $\mathcal{O}_{H_2}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_3}(\rho_j)$. Subsequently, $\mathcal{B}$ randomly selects $v_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, v_j]$ into L_{H_2}, and returns $\hat{g}^{v_j}$.
- $\mathcal{O}_{H_3}(\rho_j)$: Suppose ρ_j as the j-th different query. When $j \neq j^*$, $\mathcal{B}$ randomly selects $t_j \in \mathbb{Z}_q^*$, inserts a tuple $[\rho_j, t_j]$ into L_{H_3}, and returns $\hat{g}^{t_j}$. Otherwise, $\mathcal{B}$ inserts a tuple $[\rho_j, -]$ into L_{H_3} and returns $\hat{g}^b$.
- $\mathcal{O}_{H_4}(\eta)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_H(\eta)$. Subsequently, $\mathcal{B}$ searches for the tuple $[\eta, \Omega]$ in L_{H_4} and answers Ω.
- $\mathcal{O}_{H_5}(m,k)$: $\mathcal{B}$ randomly chooses $R \in \mathbb{Z}_q^*$, inserts a tuple $[m,k,R]$ into L_{H_5}, and answers R.
- $\mathcal{O}_{H_6}(\omega_1, \eta, C_0, C_1, C_2)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_H(\eta)$. Subsequently, $\mathcal{B}$ randomly selects $\delta \in \{0,1\}^{\lambda+l}$, inserts a tuple $[\omega_1, -, \eta, C_0, C_1, C_2, \delta]$ into L_{H_6}, and returns δ.
- $\mathcal{O}_{H_7}(m)$: $\mathcal{B}$ randomly selects $h_7 \in \hat{\mathbb{G}}$, inserts a tuple $[m, h_7]$ into L_{H_7}, and returns h_7.
- $\mathcal{O}_{H_8}(\omega_2)$: $\mathcal{B}$ randomly selects $\pi \in \hat{\mathbb{G}}$, inserts a tuple $[\omega_2, \pi]$ into L_{H_8}, and returns π.
- $\mathcal{O}_{SKGen}(\sigma_i)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\sigma_i, u_i]$ in L_{H_1}. If $i \neq i^*$, $\mathcal{B}$ returns $ek_{\sigma_i} = g^{au_i}$. Otherwise, $\mathcal{B}$ is aborted by failure.
- $\mathcal{O}_{RKGen}(\rho_j)$: $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$. There is a tuple $[\rho_j, v_j]$ in L_{H_2}. If $j \neq j^*$, $\mathcal{B}$ returns $dk_{\rho_j} = (d_1, d_2, d_3) = (\hat{g}^{at_j}, \hat{g}^{\alpha v_j}, \hat{g}^{\alpha t_j})$. Otherwise, $\mathcal{B}$ is aborted by failure.
- $\mathcal{O}_{Auth}(snd, \rho_j)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$. There is a tuple $[\rho_j, t_j]$ in L_{H_3}. When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain $dk_{\rho_j} = (d_1, d_2, d_3)$, calculates $\eta = e(H_1(\sigma_i), d_1)$, $\Omega = H_4(\eta)$ and $I = H(\eta)$, and answers $td_{(\sigma_i,\rho_j)} = Auth(pp, \sigma_i, dk_{\rho_j})$. Otherwise, $\mathcal{B}$ executes the following operations:
  - When $(i,j) \neq (i^*, j^*)$, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i}, calculates $\eta = e(ek_{\sigma_i}, H_2(\rho_j))$, $I = H(\eta)$, and $\Omega = H_4(\eta)$, calculates $d_3 = \hat{g}^{\alpha t_j}$, randomly selects $y \in \mathbb{Z}_q^*$, and returns $td_{(\sigma_i,\rho_j)} = (y_1, y_2) = (d_3^{H_4(\eta)}(\hat{f}\hat{h}^{H(\eta)})^y, \hat{g}^y)$.
  - Otherwise, $\mathcal{B}$ defines $\Omega = \Omega^*$, $I = I^*$, calculates $d_3 = \hat{g}^{b\alpha}$, randomly selects $y \in \mathbb{Z}_q^*$, and returns $td_{(\sigma_i,\rho_j)} = (y_1, y_2) = (d_3^{\Omega}(\hat{f}\hat{h}^I)^y, \hat{g}^y)$.

- $\mathcal{O}_{Enc}(\sigma_i, rcv, m)$: Let $rcv = \rho_j$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_1}(\sigma_i)$ and $\mathcal{O}_{H_2}(\rho_j)$. When $i \neq i^*$, $\mathcal{B}$ can query $\mathcal{O}_{SKGen}(\sigma_i)$ to obtain ek_{σ_i} and returns $C = Enc(pp, ek_{\sigma_i}, \rho_j, m)$. Otherwise, $\mathcal{B}$ executes the following operations:
  - When $(i,j) \neq (i^*, j^*)$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain $dk_{\rho_j} = (d_1, d_2, d_3)$, randomly selects $r \in \mathbb{Z}_q^*$, $k \in \{0,1\}^l$, calculates $R = H_5(m,k)$, $\eta = e(H_1(\sigma_i), d_1)$, $\Omega = H_4(\eta)$, $\omega_1 = e(g_1, H_2(\rho_j))^{r \cdot \Omega}$ and $\omega_2 = e(g_1, H_3(\rho_j))^{r \cdot \Omega}$, and then, returns $C = (C_0, C_1, C_2, C_3, C_4)$, where $C_0 = g^R$, $C_1 = g^r$, $C_2 = (fh^I)^r$, $C_3 = (m \parallel k) \oplus H_6(\omega_1)$, $C_4 = H_7(m)^R \cdot H_8(\omega_2)$.
  - Otherwise, $\mathcal{B}$ defines $\Omega = \Omega^*$, $I = I^*$, randomly picks $r \in \mathbb{Z}_q^*$, $\delta \in \{0,1\}^{\lambda+l}$, $k \in \{0,1\}^l$, calculates $R = H_5(m,k)$, $\omega_1 = e(g_1, H_2(\rho_j))^{r \cdot \Omega}$ and $\omega_2 = e(g_1, H_3(\rho_j))^{r \cdot \Omega}$, inserts a tuple $[\omega_1, (i,j), -, C_0, C_1, C_2, \delta]$ into L_{H_6}, and then, returns $C = (C_0, C_1, C_2, C_3, C_4)$, where $C_0 = g^R$, $C_1 = g^r$, $C_2 = (fh^I)^r$, $C_3 = (m \parallel k) \oplus \delta$, and $C_4 = H_7(m)^R \cdot H_8(\omega_2)$.
- $\mathcal{O}_{Dec}(\rho_j, snd, C)$: Let $snd = \sigma_i$. $\mathcal{B}$ performs a simulation algorithm to query $\mathcal{O}_{H_2}(\rho_j)$ and $\mathcal{O}_{H_1}(\sigma_i)$. When $j \neq j^*$, $\mathcal{B}$ can query $\mathcal{O}_{RKGen}(\rho_j)$ to obtain dk_{ρ_j} and returns the outcome of the algorithm $Dec(pp, dk_{\rho_j}, \sigma_i, C)$. Otherwise, $\mathcal{B}$ executes the following operations:
  - When $(i,j) \neq (i^*, j^*)$, $\mathcal{B}$ can query $\mathcal{O}_{Auth}(\sigma_i, \rho_j)$ to obtain $td_{(\sigma_i, \rho_j)} = (y_1, y_2)$, calculates $\omega_2 = \frac{e(C_1, y_1)}{e(C_2, y_2)}$, obtains ek_{σ_i} by querying $\mathcal{O}_{SKGen}(\sigma_i)$, calculates $\eta = e(ek_{\sigma_i}, H_3(\rho_j))$, $\Omega = H_4(\eta)$, $d_2 = H_2(\rho_j)^\alpha$, $\omega_1 = e(C_1, d_2^\Omega)$, recovers $m \parallel k$ by computing $C_3 \oplus H_6(\omega_1, \eta, C_0, C_1, C_2)$, calculates $R = H_5(m,k)$. If $C_0 = g^R$ and $C_4 = H_7(m)^R \cdot H_8(\omega_2)$ hold, $\mathcal{B}$ answers m; otherwise, $\mathcal{B}$ answers $\perp$.
  - Otherwise, $\mathcal{B}$ defines $\Omega = \Omega^*$, $I = I^*$, calculates $\omega_1 = e(C_1, \hat{g}^{v_j})^\Omega$, obtains $td_{(\sigma_i, \rho_j)} = (y_1, y_2)$ by querying $\mathcal{O}_{Auth}(\sigma_i, \rho_j)$, calculates $\omega_2 = \frac{e(C_1, y_1)}{e(C_2, y_2)}$, and searches for the corresponding tuple $[\omega_1, (i,j), -, C_0, C_1, C_2, \delta]$ in L_{H_6}. If there exists no such tuple in L_{H_6}, $\mathcal{B}$ randomly selects $\delta \in \{0,1\}^{\lambda+l}$and inserts $[\omega_1, (i,j), -, C_0, C_1, C_2, \delta]$ into L_{H_6}. Afterwards, $\mathcal{B}$ recovers $m \parallel k$ by computing $C_3 \oplus \delta$ and calculates $R = H_5(m,k)$. If $C_0 = g^R$ and $C_4 = H_7(m)^R \cdot H_8(\omega_2)$ hold, $\mathcal{B}$ answers m; otherwise, $\mathcal{B}$ answers $\perp$.

(3) *Forgery:* $\mathcal{A}_4$ outputs a triple (snd^*, ρ^*, C^*), where $snd^* = \sigma^*$ and $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$.

□

Analysis: It is obvious that the simulations of $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_5}$, $\mathcal{O}_{H_7}$, and $\mathcal{O}_{H_8}$ are perfect. Define $\eta^* = e(g, \hat{g})^{abc}$. Denote the query $\mathcal{O}_H(\eta^*)$ as the event $AskH$. Denote the failure of $\mathcal{B}$ to decrypt the legitimate ciphertext in $\mathcal{O}_{Dec}$ as the event $Derr$. Thus, $\Pr[Derr] \leq \frac{q_D}{2^{\lambda+l}}$.

Suppose E as the event for which $\sigma^* = \sigma_{i^*}$, $\rho^* = \rho_{j^*}$, and (σ^*, ρ^*, C^*) are legitimate. With ϵ and the lemma on the relationship between the chosen-identity attack and given identity attack [33], we obtain $\Pr[E] \geq \epsilon(1 - \frac{1}{q}) \frac{1}{q_{H_1} q_{H_3}}$.

Define $E_0 = AskH \vee Derr$. There is no greater over $\frac{1}{2^\lambda}$ advantage that $\mathcal{A}_4$ will forge a valid $(\sigma_{i^*}, \rho_{j^*}, C^*)$ when E_0 does not happen because $\mathcal{O}_H$, $\mathcal{O}_{H_4}$, and $\mathcal{O}_{H_6}$ are random oracles. Hence, $\Pr[E|\neg E_0] = \frac{1}{2^\lambda}$. We obtain

$$\Pr[E] \leq \Pr[E|\neg E_0]\Pr[\neg E_0] + \Pr[E_0] \leq \frac{1}{2^\lambda}(1 - \Pr[E_0]) + \Pr[E_0] = \frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda})\Pr[E_0] \leq \frac{1}{2^\lambda} + \Pr[E_0].$$

Therefore, we obtain

$$\Pr[E_0] = \Pr[AskH \vee Derr] = \Pr[AskH] + \Pr[Derr] \geq \Pr[E] - \frac{1}{2^\lambda}.$$

Subsequently, we obtain

$$\Pr[AskH] \geq \epsilon(1-\frac{1}{q})\frac{1}{q_{H_1}q_{H_3}} - \frac{1}{2^\lambda} - \Pr[Derr] \geq \epsilon(1-\frac{1}{q})\frac{1}{q_{H_1}q_{H_3}} - \frac{1+q_D}{2^\lambda}.$$

Obviously, when $AskH$ occurs, the Gap-BDH assumption can certainly be addressed by $\mathcal{B}$. $\mathcal{B}$ addresses the Gap-BDH assumption with advantage $\epsilon' = \Pr[\mathcal{B}\ success] = \Pr[AskH] \geq \epsilon(1-\frac{1}{q})\frac{1}{q_{H_1}q_{H_3}} - \frac{1+q_D}{2^\lambda}$.

6. Performance Evaluation

We first give the functionality and security comparisons, then give the comparisons of the computational overhead and communication overhead.

In Table 1, we compare our proposed IBME-ET with the related schemes (i.e., IB-ME [22], IBEET [3,15], and IBSC-ET [7]) in terms of functionality and security. It can be seen that the IB-ME scheme in [22] ensures the confidentiality, authenticity, and anonymity of data stored in the cloud, but does not achieve CCA security nor provide equality test functionality without losing the confidentiality, authenticity, and anonymity of the data. The IBEET schemes in [3,15] ensure the confidentiality of the data, but neither offer the authenticity and anonymity of data, nor provide equality test functionality without losing the confidentiality, authenticity, and anonymity of the data. Moreover, although the scheme in [3] was the first proposed IBEET scheme, it fails to achieve CCA security. Hence, the IBEET scheme that achieves CCA security was proposed in [15]. The IBSC-ET scheme in [7] ensures the confidentiality and authenticity the data and achieves CCA security, but neither ensures the anonymity of data, nor provides the equality test functionality without losing the confidentiality, authenticity, and anonymity of the data. As a result, only our proposed IBME-ET can realize all the functionality and security, which not only ensures the confidentiality, authenticity, and anonymity of the data stored in the cloud and achieves CCA security, but also provides equality test functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data.

Table 1. Comparison of functionality and security.

	Equality Test	**Confidentiality**	**Authenticity**	**Anonymity**	
				Sender	**Receiver**
[22]	✗	CPA	✓	✓	✓
[3]	✓	CPA	✗	✗	✗
[15]	✓	CCA	✗	✗	✗
[7]	✓	CCA	✓	✗	✗
Ours	✓	CCA	✓	✓	✓

Note that the IB-ME scheme in [22] implements only CPA security. This means that the ciphertexts are malleable. When a valid plaintext/ciphertext pair of the sender and receiver is given, an attacker can utilize it to fake a valid ciphertext of any message, in this way to break the authenticity of the ciphertext stored in the cloud. Moreover, the IB-ME scheme in [22] cannot provide equality test functionality for ciphertexts. Obviously, the IB-ME scheme in [22] is not applicable to cloud storage application scenarios. In addition, it was proven in [15] that the computational overhead and communication overhead of the IBEET scheme in [15] are comparable to those of the IBEET scheme in [3]; however, the IBEET scheme in [15] achieves stricter CCA security while the IBEET scheme in [3] only achieves CPA security. Therefore, we only compared our proposed IBME-ET with the most-related schemes (i.e., IBEET [15] and IBSC-ET [7]) in terms of computational overhead and communication overhead.

Table 2 shows the computational overhead comparison, which theoretically analyzes the computational cost of our proposed scheme and the comparative schemes with regard to encryption key generation (indicated as SKGen), decryption key generation (indicated as RKGen), encryption (indicated as Enc), decryption (indicated as Dec), authorization (indicated as Auth), and the equality test (indicated as Test). For the analysis, we concentrated on the operations that consumed the most time, including hash-to-point, bilinear pairing, and exponentiation. Notably, the authorization algorithms of the schemes in [7,15] have no computational cost. This is because both schemes directly use the partial decryption private key as the trapdoor regardless of anonymity. The communication overhead comparison is given in Table 3, which theoretically analyzes the communication cost of our proposed scheme and the comparative schemes with regard to the encryption private key, decryption private key, trapdoor, and ciphertext.

Table 2. Comparison of computational overhead.

	SKGen	**RKGen**	**Enc**	**Dec**	**Auth**	**Test**
[15]	-	$3\hat{h}+3\hat{e}$	$3\hat{h}+3p+6e$	$3p+2e$	0	$2p+2e$
[7]	$2h+2e$	$2\hat{h}+2\hat{e}$	$2\hat{h}+2p+5e+\hat{e}$	$2h+5p+2e+\hat{e}$	0	$4p$
Ours	$h+e$	$2\hat{h}+3\hat{e}$	$2\hat{h}+3p+5e+\hat{e}$	$h+3p+2e+\hat{e}$	$h+p+4\hat{e}$	$6p$

$e, \hat{e}$ are exponentiation operations in $\mathbb{G}$ and $\hat{\mathbb{G}}$, respectively. $h, \hat{h}$ are hash-to-point operations in $\mathbb{G}$ and $\hat{\mathbb{G}}$, respectively. p is the pairing operation.

Table 3. Comparison of communication overhead.

	Encryption Key	**Decryption Key**	**Trapdoor**	**Ciphertext**
[15]	-	$3\lvert\hat{\mathbb{G}}\rvert$	$\lvert\hat{\mathbb{G}}\rvert$	$4\lvert\mathbb{G}\rvert+5\lambda$
[7]	$2\lvert\mathbb{G}\rvert$	$2\lvert\hat{\mathbb{G}}\rvert$	$\lvert\hat{\mathbb{G}}\rvert$	$3\lvert\mathbb{G}\rvert+\lvert\hat{\mathbb{G}}\rvert+\lvert\mathbb{Z}_q\rvert+\lambda$
Ours	$\lvert\mathbb{G}\rvert$	$3\lvert\hat{\mathbb{G}}\rvert$	$2\lvert\hat{\mathbb{G}}\rvert$	$3\lvert\mathbb{G}\rvert+\lvert\hat{\mathbb{G}}\rvert+\lvert\mathbb{Z}_q\rvert+\lambda$

$\lvert\mathbb{G}\rvert, \lvert\hat{\mathbb{G}}\rvert$ are the sizes of the elements in groups $\mathbb{G}$ and $\hat{\mathbb{G}}$, respectively. $\lvert\mathbb{Z}_q\rvert$ is the size of the elements in $\mathbb{Z}_q$, and λ is the security level.

In order to compare the computational and communication overhead of our proposed scheme with the comparative schemes more intuitively, we used Charm 0.50 in Python 3.6.9 to implement these schemes. The experimental environment was configured as follows: Intel(R) Xeon(R) Platinum 8124M CPU @ 2.70 GHz (Intel Corporation, Santa Clara, CA, USA), 16 GB memory, and Ubuntu 18.03 LTS. The experiments were instantiated using the MNT224 curve in Charm and employed the Python module *timeit* for the time measurements. Figure 3 shows the experimental computational overheads of these schemes, and Figure 4 shows the experimental communication overheads of these schemes.

From Tables 1–3 and Figures 3 and 4, we can conclude that, with a small sacrifice in computational and communication efficiency, our IBME-ET scheme not only offers the confidentiality, authenticity, and anonymity of the data and achieves CCA security, but also provides equality test functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data. Other related schemes cannot support this feature.

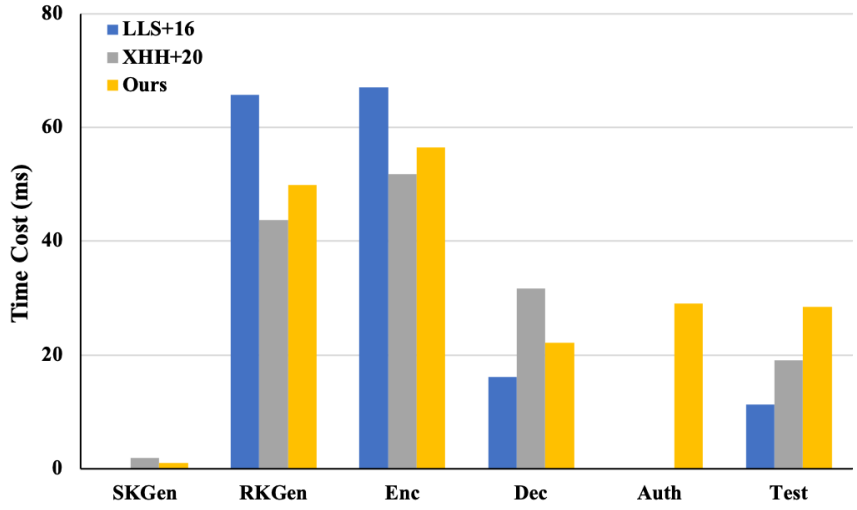

Figure 3. Computational overhead comparison with LLS+16 [15] and XHH+20 [7].

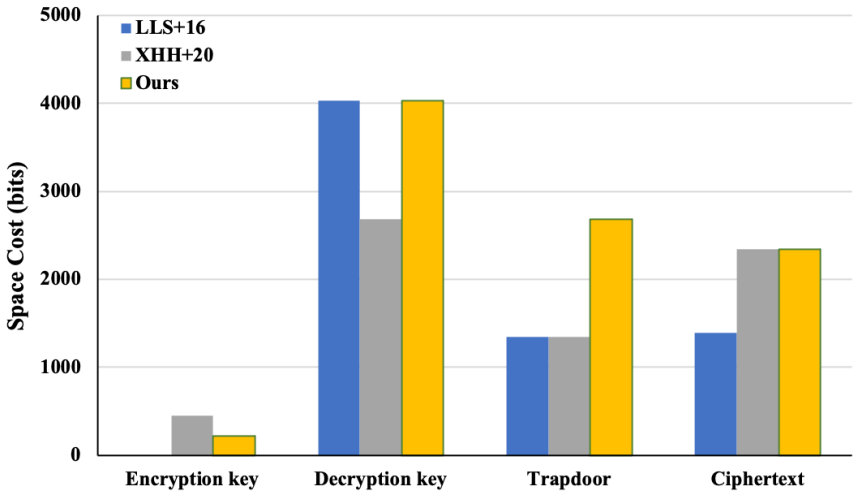

Figure 4. Communication overhead comparison with LLS+16 [15] and XHH+20 [7].

7. Conclusions

In this paper, we presented the primitive of the IBME-ET, which not only offers the confidentiality, authenticity, and anonymity of data and achieves CCA security, but also provides equality test functionality for ciphertexts generated under different identities without losing the confidentiality, authenticity, and anonymity of the data. More precisely, we introduced the system model and definition of the IBME-ET. With respect to the confidentiality, authenticity, and anonymity, we formalized the security models for the IBME-ET. Finally, we proposed a concrete IBME-ET scheme, and our scheme was confirmed to be secure and practical by proving its security and evaluating its performance.

Author Contributions: Conceptualization, Z.Y. and X.L.; methodology, Z.Y.; validation, H.Q. and J.X.; writing—original draft, Z.Y. and X.L.; writing—review and editing, H.Q. and X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Boneh, D.; Di Crescenzo, G.; Ostrovsky, R.; Persiano, G. Public key encryption with keyword search. In Proceedings of the Advances in Cryptology—EUROCRYPT 2004, Interlaken, Switzerland, 2–6 May 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 506–522.
2. Yang, G.; Tan, C.H.; Huang, Q.; Wong, D.S. Probabilistic public key encryption with equality test. In Proceedings of the Topics in Cryptology—CT-RSA 2010, San Francisco, CA, USA, 1–5 March 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 119–131.
3. Ma, S. Identity-based encryption with outsourced equality test in cloud computing. *Inf. Sci.* **2015**, *328*, 389–402. [CrossRef]
4. Lu, J.; Li, H.; Huang, J.; Ma, S.; Au, M.H.A.; Huang, Q. An Identity-Based Encryption with Equality Test scheme for healthcare social apps. *Comput. Stand. Interfaces* **2023**, *87*, 103759. [CrossRef]
5. *My HealtheVet*. Available online: http://www.myhealth.va.gov (accessed on 22 December 2023).
6. Vaanchig, N.; Qin, Z.; Ragchaasuren, B. Constructing secure-channel free identity-based encryption with equality test for vehicle-data sharing in cloud computing. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e3896. [CrossRef]
7. Xiong, H.; Hou, Y.; Huang, X.; Zhao, Y. Secure message classification services through identity-based signcryption with equality test towards the Internet of vehicles. *Veh. Commun.* **2020**, *26*, 100264. [CrossRef]
8. Ohtaki, Y. Constructing a Searchable Encrypted Log Using Encrypted Inverted Indexes. In Proceedings of the 2005 International Conference on Cyberworlds, CW 2005, Singapore, 23–25 November 2005; pp. 130–138.
9. Boneh, D.; Kushilevitz, E.; Ostrovsky, R.; Skeith, W.E. Public key encryption that allows PIR queries. In Proceedings of the Advances in Cryptology—CRYPTO 2007, Santa Barbara, CA, USA, 19–23 August 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 50–67.
10. Camenisch, J.; Kohlweiss, M.; Rial, A.; Sheedy, C. Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data. In Proceedings of the Public Key Cryptography—PKC 2009, Irvine, CA, USA, 18–20 March 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 196–214.
11. Curtmola, R.; Garay, J.A.; Kamara, S.; Ostrovsky, R. Searchable symmetric encryption: Improved definitions and efficient constructions. In Proceedings of the ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, 30 October–3 November 2006; pp. 79–88.
12. Cash, D.; Jarecki, S.; Jutla, C.S.; Krawczyk, H.; Rosu, M.; Steiner, M. Highly-scalable searchable symmetric encryption with support for Boolean queries. In Proceedings of the Advances in Cryptology—CRYPTO 2013, Santa Barbara, CA, USA, 18–22 August 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 353–373.
13. Tang, Q. Public key encryption supporting plaintext equality test and user-specified authorization. *Secur. Commun. Netw.* **2012**, *5*, 1351–1362. [CrossRef]
14. Ma, S.; Huang, Q.; Zhang, M.; Yang, B. Efficient public key encryption with equality test supporting flexible authorization. *IEEE Trans. Inf. Forensic Secur.* **2014**, *10*, 458–470. [CrossRef]
15. Lee, H.T.; Ling, S.; Seo, J.H.; Wang, H. Semi-generic construction of public key encryption and identity-based encryption with equality test. *Inf. Sci.* **2016**, *373*, 419–440. [CrossRef]
16. Lin, X.J.; Sun, L.; Qu, H. Generic construction of public key encryption, identity-based encryption and signcryption with equality test. *Inf. Sci.* **2018**, *453*, 111–126. [CrossRef]
17. Li, N. Efficient equality test on identity-based ciphertexts supporting flexible authorization. *Entropy* **2023**, *25*, 362–379. [CrossRef] [PubMed]
18. Boyen, X. Multipurpose Identity-Based Signcryption. In Proceedings of the Advances in Cryptology—CRYPTO 2003, Santa Barbara, CA, USA, 17–21 August 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 383–399.
19. Xiong, H.; Zhao, Y.; Hou, Y.; Huang, X.; Jin, C.; Wang, L.; Kumari, S. Heterogeneous Signcryption With Equality Test for IIoT Environment. *IEEE Internet Things J.* **2021**, *8*, 16142–16152. [CrossRef]
20. Xiong, H.; Hou, Y.; Huang, X.; Zhao, Y.; Chen, C.M. Heterogeneous Signcryption Scheme from IBC to PKI with Equality Test for WBANs. *IEEE Syst. J.* **2022**, *16*, 2391–2400. [CrossRef]
21. Hou, Y.; Huang, X.; Chen, Y.; Kumar, S.; Xiong, H. Heterogeneous signcryption scheme supporting equality test from PKI to CLC toward IoT. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4190. [CrossRef]
22. Ateniese, G.; Francati, D.; Nuñez, D.; Venturi, D. Match Me if You Can: Matchmaking Encryption and Its Applications. In Proceedings of the Advances in Cryptology—CRYPTO 2019, Santa Barbara, CA, USA, 18–22 August 2019; Springer: Cham, Switzerland, 2019; pp. 701–731.
23. Xu, S.; Ning, J.; Li, Y.; Zhang, Y.; Xu, G.; Huang, X.; Deng, R.H. Match in my way: Fine-grained bilateral access control for secure cloud-fog computing. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 1064–1077. [CrossRef]
24. Sun, J.; Yuan, Y.; Tang, M.; Cheng, X.; Nie, X.; Aftab, M.U. Privacy-preserving bilateral fine-grained access control for cloud-enabled industrial IOT healthcare. *IEEE Trans. Ind. Inform.* **2021**, *18*, 6483–6493. [CrossRef]
25. Chen, J.; Li, Y.; Wen, J.; Weng, J. Identity-Based Matchmaking Encryption from Standard Assumptions. In Proceedings of the Advances in Cryptology—ASIACRYPT 2022, Taipei, Taiwan, 5–9 December 2022; Springer: Cham, Switzerland, 2022; pp. 394–422.
26. Wu, A.; Luo, W.; Weng, J.; Yang, A.; Wen, J. Fuzzy Identity-Based Matchmaking Encryption and Its Application. *IEEE Trans. Inf. Forensic Secur.* **2023**, *18*, 5592–5607. [CrossRef]
27. Yan, Z.; Qu, H.; Zhang, X.; Xu, J.L.; Lin, X.J. Identity-based proxy matchmaking encryption for cloud-based anonymous messaging systems. *J. Syst. Archit.* **2023**, *142*, 102950. [CrossRef]

28. Sun, J.; Xu, G.; Zhang, T.; Yang, X.; Alazab, M.; Deng, R.H. Privacy-Aware and Security-Enhanced Efficient Matchmaking Encryption. *IEEE Trans. Inf. Forensic Secur.* **2023**, *18*, 4345–4360. [CrossRef]
29. Boyen, X. A tapestry of identity-based encryption: Practical frameworks compared. *Int. J. Appl. Cryptogr.* **2008**, *1*, 3–21. [CrossRef]
30. Bellare, M.; Rogaway, P. Random oracles are practical: A paradigm for designing efficient protocols. In Proceedings of the ACM Conference on Computer and Communications Security, CCS 1993, Fairfax, VA, USA, 3–5 November 1993; ACM: New York, NY, USA, 1993; pp. 62–73.
31. Tibouchi, M. *Encyclopedia of Cryptography and Security*; Springer: Boston, MA, USA, 2011.
32. Franklin, J. *Proof in Mathematics: An Introduction*; Quakers Hill Press: Sydney, Australia, 1996.
33. Choon, J.C.; Hee Cheon, J. An identity-based signature from gap Diffie-Hellman groups. In Proceedings of the 6th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2002, Miami, FL, USA, 6–8 January 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 18–30.