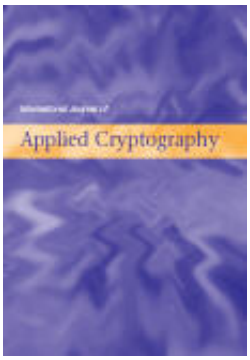


International Journal of Applied Cryptography

ISSN online: 1753-0571 - ISSN print: 1753-0563
https://www.inderscience.com/ijact

Efficient revocable identity-based encryption with equality test

Jiaojiao Du, Sha Ma, Tian Yang, Qiong Huang

DOI: 10.1504/IJACT.2023.10060444

Article History:

Received:	01 December 2022
Last revised:	26 May 2023
Accepted:	10 June 2023
Published online:	03 May 2024

Efficient revocable identity-based encryption with equality test

Jiaojiao Du, Sha Ma*, Tian Yang and Qiong Huang

College of Mathematics and Informatics,
South China Agricultural University,
Guangzhou, Guangdong, China
Email: dujiaojiao@stu.scau.edu.cn
Email: martin_deng@scau.edu.cn
Email: 20203162056@stu.scau.edu.cn
Email: qhuang@scau.edu.cn
*Corresponding author

Abstract: Identity-based encryption with equality test (IBEET) provides an attractive method to test whether two ciphertexts are encryptions of the same plaintext without certificate managements. However, none of the existing IBEET constructions can provide a way to revoke the user in the scenario where the user's private key is compromised or identity gets expired, which is undesirable for identity-based setting. Furthermore, the user cannot revoke the tester when it no longer wants the tester to test its ciphertexts. How to achieve both user and tester revocation in IBEET remains a challenging task. In this paper, we propose a new primitive called revocable identity-based encryption with equality test (R-IBEET), which can solve the aforementioned two problems simultaneously, and formalise the security models of R-IBEET against three types of adversaries. Then we propose a concrete construction of R-IBEET. Our scheme is pairing-free, thus is more efficient compared with the related work.

Keywords: cloud computing; identity-based encryption; equality test; revocation.

Reference to this paper should be made as follows: Du, J., Ma, S., Yang, T. and Huang, Q. (2023) 'Efficient revocable identity-based encryption with equality test', *Int. J. Applied Cryptography*, Vol. 4, Nos. 3/4, pp.205–218.

Biographical notes: Jiaojiao Du received her MS degree from the College of Informatics, South China Agricultural University, Guangzhou, China. Her research interests include information security and cryptography.

Sha Ma received her MS from the Wuhan University, Wuhan, China in 2006 and her PhD from the South China Agricultural University, Guangzhou, China in 2012, where she is currently an Associate Professor at the College of Informatics. Her research interests include applied cryptography and security in cloud computing.

Tian Yang received his MS from the College of Informatics, South China Agricultural University, Guangzhou, China. His research interests include public key cryptography and information security.

Qiong Huang received his MS from the Fudan University, Shanghai, China in 2006 and his PhD from the City University of Hong Kong, Hong Kong in 2010. He is currently a Professor at the College of Informatics, South China Agricultural University, Guangzhou, China. His research interests include cryptography and information security, in particular, cryptographic protocols design and analysis.

1 Introduction

As cloud storage has grown by leaps and bounds in the current big data era, data users can save local storage as well as simplify data management by outsourcing their data to the cloud. To avoid unauthorised use and disclosure of some highly sensitive information, users usually store their data in form of ciphertexts on cloud. However, this causes new problems for data search owing to the structural changes in encrypted data. To solve the above-mentioned issue, Boneh et al. (2004) creatively put forward a notion called public key encryption with keyword search (PEKS). In PEKS, the data user can compute a trapdoor for a keyword using its private key, and sends it to the cloud server whenever it wants to retrieve documents that contain the keyword. With the trapdoor, the cloud server

is able to check whether the ciphertext is the encryption of the keyword in the issued trapdoor. PEKS is a useful cryptographic tool and it has been applied in many fields such as the intelligent 6G wireless systems (Wang et al., 2023), community segmented vehicular social networks (Khowaja et al., 2023) and internet of medical things (Deebak et al., 2022). Nonetheless, PEKS is only feasible for checking equality between ciphertexts that are generated from the identical public key.

In Cryptographers' Track at the RSA Conference (CT-RSA), Yang et al. (2010) first proposed public key encryption with equality test (PKEET), which allows anyone to perform equality test on two ciphertexts that are generated from different public keys. However, due to the intricate certificate management problem, the proposed scheme does not scale well. By combining identity-based cryptosystem with PKEET, Ma (2016) presented identity-based encryption with equality test (IBEET), which solves the certificate management problem in PKEET. Based on Ma's work, various subsequent IBEET schemes (Lin et al., 2018, 2021b; Ming and Wang, 2019; Ramadan et al., 2020; Alornyo et al., 2020; Susilo et al., 2020) have been presented.

User revocation is an important problem in identity-based cryptosystem. However, there are no existing IBEET constructions that can achieve user revocation. Without user revocation mechanism, potential attacks may be mounted by a malicious user in the scenario where the user's private key is compromised or identity gets expired. That is, when a user's private key is compromised or identity gets expired, it should no longer be able to decrypt the ciphertexts. User revocation mechanism guarantees that user can do decryption only under the condition that it is not revoked. Besides the issue of user revocation, we notice that in the related works of IBEET, the tester acquires permanent test right after being authorised, and it cannot be revoked when the user no longer wants it to perform equality test. Inspired by the work of Sun et al. (2020), we resolve the problem of user and tester revocation through introducing time keys into our scheme, and present a notion called revocable identity-based encryption with equality test (R-IBEET). In our scheme, decryption and test keys are all time related. When user or tester needs to be revoked, they will not be able to obtain the current time keys and the ciphertexts will be updated to new ones by the cloud server. Due to the lack of the current time keys, the revoked user and tester are unable to complete decryption and equality test separately, thus achieving revocation for both user and tester. Using this way, the ciphertexts will not become longer and longer with the number of updates, and the cloud server only need to keep the current ciphertexts, therefore, it will help to save storage resource.

1.1 Related work

1.1.1 Public key encryption with equality test

In CT-RSA 2010, Yang et al. (2010) first proposed the primitive PKEET, which allows anyone to check whether two different ciphertexts are encryptions of the same plaintext. However, without authorisation mechanism, anyone can perform equality test without limitation, and hence causes potential risk to user's privacy. To overcome this problem, Tang (2011) proposed the conception of PKEET with fine-grained authorisation (PKEET-FG). Subsequently, Tang (2012) presented an all-or-nothing PKEET (AoN-PKEET) scheme, where a proxy can perform equality test only if it is authorised by the user. To strengthen data privacy of the user, Ma et al. (2015) presented a PKEET scheme with flexible authorisation (PKEET-FA), which supports four kinds of authorisations. Lin et al. (2021a) optimised Ma et al.'s scheme and proposed a new pairing-free PKEET-FA scheme. However, all the aforementioned schemes suffer from the intricate certificate management problem.

1.1.2 Identity-based encryption with equality test

To solve the certificate management problem in traditional public key cryptography, Ma (2016) integrated identity-based cryptosystem into PKEET and proposed the primitive IBEET in 2016. However, due to the use of time-consuming operations like bilinear pairing, Ma's scheme is not efficient enough. Later, Wu et al. (2017) and Wu et al. (2018) improved the efficiency of Ma's work. To further enhance the efficiency, Wu et al. (2019) presented an efficient IBEET scheme without pairings in 2019.

1.1.3 Revocable encryption

Revocation mechanism has been widely studied in PEKS and identity-based encryption (IBE). In 2005, Abdalla et al. first proposed a temporarily searchable encryption scheme where the trapdoor is only issued for some desired time. Yu et al. (2014) constructed a new revocable PEKS scheme in 2013. In their scheme, the cloud server can perform equality test only when it is not revoked. Zhang and Mao (2016) employed hash chain to construct another revocable PEKS scheme that can resist off-line keyword guessing attacks.

Boneh and Franklin (2001) presented the first revocable IBE construction in 2001. However, their scheme is not scalable as the private key generator (PKG) needs to deliver new private keys for all the non-revoked users per each time period. Using complete subtree structure, Boldyreva et al. (2008) constructed the first scalable IBE scheme which reduces the complexity of updating key from linear to logarithmic. Following the work of Boldyreva et al. (2008), many revocable IBE schemes (Libert et al., 2009; Li et al., 2015; Qin et al., 2015; Wei et al., 2016) have been proposed later on. Sun et al. (2018) presented a revocable IBE scheme with ciphertext evolution in the cloud. In their scheme, PKG periodically delivers time keys to non-revoked users as well as the cloud server. This work was extended to broadcast setting in Sun et al. (2020).

1.2 Contribution

Below, we will list the contributions of our study.

1 In this paper, we first introduce revoke mechanism into IBEET and propose a notion called revocable identity-based encryption with equality test (R-IBEET). R-IBEET can achieve both user and tester revocation according to the practical needs.

2 We prove that our R-IBEET scheme can achieve OW-sID-CCA security against type-I adversary, OW-sID-CPA security against type-II adversary, as well as IND-sID-CCA security against type-III adversary.

3 Taking advantage of Shamir's secret sharing, we propose an efficient R-IBEET scheme without bilinear pairings. Our scheme enjoys especially high test efficiency compared with the related work.

2 Preliminaries

2.1 Computational Diffie-Hellman (CDH) problem:

Given $(g, g^a, g^c) \in \mathbb{G}^3$ for $a, c \in \mathbb{Z}_p$, where $\mathbb{G}$ is a cyclic group of prime order p, and g is a random generator of $\mathbb{G}$. We define the advantage of any probabilistic polynomial-time algorithm $\mathcal{A}$ in computing g^{ac} as

$$\mathbf{Adv}^{\mathrm{CDH}}_{\mathcal{A},\mathbb{G}} \stackrel{\mathrm{def}}{=} \Pr[\mathcal{A}(g, g^a, g^c) = g^{ac}].$$

The CDH assumption holds provided that $\mathbf{Adv}^{\mathrm{CDH}}_{\mathcal{A},\mathbb{G}}$ is negligible.

2.2 Shamir's secret sharing

Shamir (1979) first proposed a secret sharing scheme based on Lagrange difference polynomials, and the detail is as follows:

Let $(x_1, y_1), \ldots, (x_t, y_t)$ be t different points, these t points can uniquely determine a polynomial $f(x)$ of degree $(t-1)$ satisfying $f(x_i) = y_i$ for $1 \le i \le t$, where

$$f(x) = \sum_{i=1}^{t} f(x_i) \prod_{1 \le j \ne i \le t} \frac{x - x_j}{x_i - x_j}.$$

Let $s \in \mathbb{Z}_p$ be a secret satisfying $s = f(0) = a_0$. Choose uniform $a_1, \ldots, a_t \in \mathbb{Z}_p$, and then compute the polynomial $f(x_i) = s + \sum_{i=1}^{t} a_i x^i$. Each share of s is denoted as $(x_i, f(x_i))$ for $1 \le i \le n$. The secret s can be reconstructed by pooling any t shares together as below

$$s = f(0) = \sum_{i=1}^{t} f(x_i) \prod_{1 \le j \ne i \le t} \frac{x_j}{x_j - x_i}.$$

3 Definition

3.1 System model

The system model of R-IBEET is shown in Figure 1. There are four participants and the detail is depicted as follows:

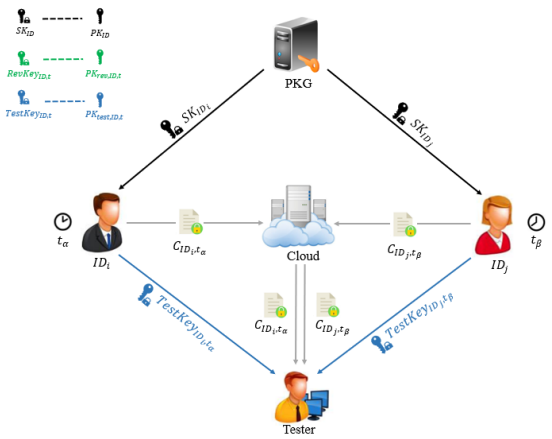
1 *Private key generator (PKG):* it can generate users' private/public key pairs and issue revoke keys for non-revoked users, as well as issue evolve keys for the cloud server to evolve the ciphertexts.

2 *User:* the user registers itself at PKG and issues test key for the tester to do equality test on its ciphertexts, as well as issues test key for the PKG to generate the evolve key.

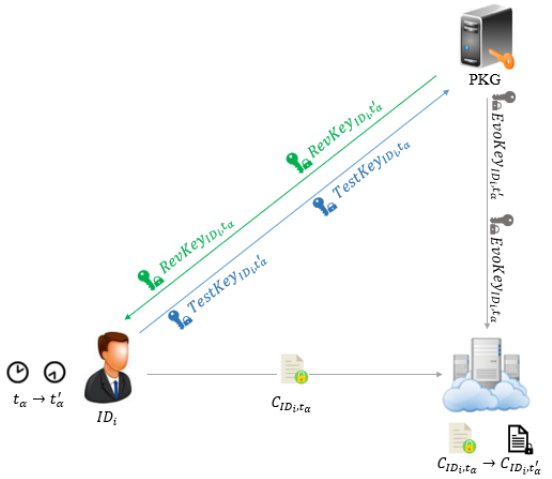
3 *Tester:* it is responsible for doing equality test with test key and users' ciphertexts.

4 *Cloud server:* when the user or the tester need to be revoked, it will update the user's ciphertexts stored on it.

Figure 1 System model of our R-IBEET scheme (see online version for colours)



(a)



(b)

3.2 Algorithms

Definition 3.1: A R-IBEET scheme is made up of the following ten algorithms: Setup, Extract, TimeKeyTest, TimeKeyRev, TimeKeyEvo, Encrypt, Decrypt, Test, Revoke, CiphertextEvolve, where $\mathcal{M}$ is its plaintext space and $\mathcal{C}$ is its ciphertext space.

1 Setup(λ): given a security parameter λ, this algorithm outputs a public parameter $parm$ and a master secret key msk.

2 Extract(msk, ID): given the public parameter $parm$, the master secret key msk and an identity ID, this algorithm outputs a private key SK_{ID} and the corresponding public key PK_{ID} for that identity, which are sent to the user. It is run by the PKG.

3 TimeKeyTest(SK_{ID}, t): given a private key SK_{ID} and a time tag t, this algorithm outputs a test key $TestKey_{ID,t}$ and the corresponding public test key $PK_{test,ID,t}$, which are sent to the tester and the PKG. It is run by the user.

4 TimeKeyRev(msk, ID, t): given the master secret key msk, an identity ID and a time tag t, this algorithm outputs a revoke key $RevKey_{ID,t}$ and the corresponding public revoke key $PK_{rev,ID,t}$, which are sent to the user. It is run by the PKG.

5 TimeKeyEvo($msk, ID, t, TestKey_{ID,t}$): given the master secret key msk, an identity ID, a time tag t and a test key $TestKey_{ID,t}$, this algorithm outputs an evolve key $EvoKey_{ID,t}$, which is sent to the cloud server. It is run by the PKG.

6 Encrypt($m, PK_{ID}, PK_{rev,ID,t}, PK_{test,ID,t}$): given $m \in \mathcal{M}$, a public key PK_{ID}, a public revoke key $PK_{rev,ID,t}$ and a public test key $PK_{test,ID,t}$, this algorithm outputs a ciphertext $C_{ID,t} \in \mathcal{C}$.

7 Decrypt($C_{ID,t}, SK_{ID}, RevKey_{ID,t}$): given a ciphertext $C_{ID,t}$, a private key SK_{ID} and a revoke key $RevKey_{ID,t}$, this algorithm outputs a message m or a failure symbol $\perp$. It is run by the user.

8 Test($C_{ID_i,t_\alpha}, TestKey_{ID_i,t_\alpha}, C_{ID_j,t_\beta}, TestKey_{ID_j,t_\beta}$): given a ciphertext $C_{ID_i,t_\alpha} \in \mathcal{C}$ with time t_α of user U_i with identity ID_i, a test key $TestKey_{ID_i,t_\alpha}$ with time t_α of user U_i with identity ID_i, a ciphertext $C_{ID_j,t_\beta} \in \mathcal{C}$ with time t_β of user U_j with identity ID_j and a test key $TestKey_{ID_j,t_\beta}$ with time t_β of user U_j with identity ID_j, this algorithm outputs 1 if messages hidden within C_{ID_i,t_α} and C_{ID_j,t_β} are the same, or 0 otherwise.

9 Revoke(ID, t): given an identity ID and a time tag t, the PKG stops issuing the revoke key $RevKey_{ID,t}$ for the user with identity ID at time t.

10 CiphertextEvolve($C_{ID,t}, ID, EvoKey_{ID,t}, EvoKey_{ID,t'}$): given a ciphertext $C_{ID,t} \in \mathcal{C}$ with time t, an identity ID, two evolve keys $EvoKey_{ID,t}$ with time t, and $EvoKey_{ID,t'}$ with time t', this algorithm outputs a ciphertext $C_{ID,t'}$ with time t'. It is run by the cloud server.

3.3 Security models

We formally define the following three types of adversaries for security models of R-IBEET. These three types of adversaries are chosen based on the system model. The PKG and the sender are seen as honest, so they are not chosen to be adversaries. The revoked user and the cloud server have private key and evolve key, respectively that may be helpful to obtain plaintext from the ciphertext, so they are chosen to be adversaries. In addition to these two types of adversaries, it is generally necessary to consider the outsider, who can obtain ciphertext but does not have any key.

1 *Type-I adversary:* this adversary is a malicious revoked user who has private key and test key, but does not have revoke key and evolve key. Given the challenge ciphertext, this adversary aims to decode the plaintext hidden within.

2 *Type-II adversary:* this adversary can be the semi-trusted cloud server who has evolve key, and who might have test key as well as revoke key ('$-$' in Table 1 denotes 'might have'), but does not have user's private key. Given the challenge ciphertext, this adversary aims to decode the plaintext hidden within.

3 *Type-III adversary:* this adversary is an outsider who has none of private key, test key, revoke key and evolve key. It aims to distinguish which of the two messages is the challenge ciphertext encrypted from.

Table 1 Types of adversaries

	Private key	*Test key*	*Revoke key*	*Evolve key*
Type-I adversary	✓	✓	×	×
Type-II adversary	×	–	–	✓
Type-III adversary	×	×	×	×

3.3.1 OW-sID-CCA security against type-I adversary

$\mathbf{Exp}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}$: let $\mathcal{A}_1$ be a type-I adversary. It interacts with the challenger $\mathcal{G}$ via the following game. Init: $\mathcal{A}_1$ outputs an identity ID^* and a time tag t^* to be challenged.

1 Setup: given a security parameter λ, $\mathcal{G}$ runs the algorithm Setup to provide the public parameters $parm$ to $\mathcal{A}_1$, while keeps the master secret key msk for itself.

2 Phase 1: $\mathcal{A}_1$ can make the following queries adaptively in this phase. The constraint is that $\langle ID^*, t^* \rangle$ does not appear in any $\mathcal{O}_{RevKey}$ query and $\mathcal{O}_{EvoKey}$ query.

- $\mathcal{O}_{ExtKey}$ query $\langle ID \rangle$: $\mathcal{G}$ runs algorithm Extract to obtain the private key SK_{ID} and the

corresponding public key PK_{ID} for that identity. It sends SK_{ID} and PK_{ID} to $\mathcal{A}_1$.

- $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ runs algorithm Extract to obtain the private key SK_{ID}. It then runs algorithm TimeKeyTest to obtain the test key $TestKey_{ID,t}$ and the corresponding public test key $PK_{test,ID,t}$ using the private key SK_{ID}. It sends $TestKey_{ID,t}$ and $PK_{test,ID,t}$ to $\mathcal{A}_1$.
- $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ runs algorithm TimeKeyRev to obtain the revoke key $RevKey_{ID,t}$ and the corresponding public revoke key $PK_{rev,ID,t}$. It sends $RevKey_{ID,t}$ and $PK_{rev,ID,t}$ to $\mathcal{A}_1$.
- $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ runs algorithm Extract to obtain the private key SK_{ID}. It then runs algorithm TimeKeyTest to obtain the test key $TestKey_{ID,t}$ using the private key SK_{ID}. Then, it runs algorithm TimeKeyEvo to obtain the evolve key $EvoKey_{ID,t}$ using the test key $TestKey_{ID,t}$. It sends $EvoKey_{ID,t}$ to $\mathcal{A}_1$.
- $\mathcal{O}_{Dec}$ query $\langle C_{ID,t}, ID, t\rangle$: $\mathcal{G}$ runs algorithm Extract to obtain the private key SK_{ID}. It then runs algorithm TimeKeyRev to obtain the revoke key $RevKey_{ID,t}$. Then, it runs algorithm Decrypt to decrypt the ciphertext into a plaintext using the private key SK_{ID} and the revoke key $RevKey_{ID,t}$. Finally, return the plaintext to $\mathcal{A}_1$.

3 Challenge: $\mathcal{G}$ randomly selects a message $m \in \mathcal{M}$ and encrypts the message via algorithm Encrypt to C_{ID^*,t^*}. Finally, return the challenge ciphertext C_{ID^*,t^*} to $\mathcal{A}_1$.

4 Phase 2: $\mathcal{A}_1$ continues to issue more queries as below:

- $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $O_{TestKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle \neq \langle ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle \neq \langle ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{Dec}$ query $\langle C_{ID,t}, ID, t\rangle \neq \langle C_{ID^*,t^*}, ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.

Notice that $\langle ID^*\rangle$ could appear in $\mathcal{O}_{ExtKey}$ query and $\langle ID^*, t^*\rangle$ could appear in $\mathcal{O}_{TestKey}$ query.

4 Guess: $\mathcal{A}_1$ outputs a guess m' and wins if $m = m'$.

We define the advantage of $\mathcal{A}_1$ in the above game as

$$\mathbf{Adv}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}(\lambda) = |\Pr[m = m']|.$$

Definition 3.2 We say that a R-IBEET scheme is OW-sID-CCA secure, if for each type-I adversary, $\mathbf{Adv}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}(\lambda)$ is negligible in the security parameter λ.

3.3.2 OW-sID-CPA security against type-II adversary

$\mathbf{Exp}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}$: let $\mathcal{A}_2$ be a type-II adversary. It interacts with the challenger $\mathcal{G}$ via the following game. Init: $\mathcal{A}_2$ outputs an identity ID^* and a time tag t^* to be challenged.

1 Setup: given a security parameter λ, $\mathcal{G}$ runs the algorithm Setup to provide the public parameters $parm$ to $\mathcal{A}_2$, while keeps the master secret key msk for itself.

2 Phase 1: $\mathcal{A}_2$ can make $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle$, $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle$, $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle$, and $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle$ adaptively in this phase. The constraint is that $\langle ID^*\rangle$ does not appear in any $\mathcal{O}_{ExtKey}$ query.

3 Challenge: $\mathcal{G}$ picks a random message $m \in \mathcal{M}$ and encrypts the message via algorithm Encrypt to C_{ID^*,t^*}. Finally, return the challenge ciphertext C_{ID^*,t^*} to $\mathcal{A}_2$.

4 Phase 2: $\mathcal{A}_2$ continues to issue more queries as below:

- $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle \neq \langle ID^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle$: $\mathcal{G}$ performs as in Phase 1.

Notice that $\langle ID^*, t^*\rangle$ could appear in $\mathcal{O}_{EvoKey}$ query. To maximise capability of the type-II adversary, here we also allow $\langle ID^*, t^*\rangle$ to appear in $\mathcal{O}_{TestKey}$ query and $\mathcal{O}_{RevKey}$ query.

5 Guess: $\mathcal{A}_2$ outputs a guess m' and wins if $m = m'$.

We define the advantage of $\mathcal{A}_2$ in the above game as

$$\mathbf{Adv}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}(\lambda) = |\Pr[m = m']|.$$

Definition 3.3: We say that a R-IBEET scheme is OW-sID-CPA secure, if for each type-II adversary, $\mathbf{Adv}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}(\lambda)$ is negligible in the security parameter λ.

3.3.3 IND-sID-CCA security against type-III adversary

$\mathbf{Exp}^{\text{IND-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_3}$: let $\mathcal{A}_3$ be a type-III adversary. It interacts with the challenger $\mathcal{G}$ via the following game. Init: $\mathcal{A}_3$ outputs an identity ID^* and a time tag t^* to be challenged.

1 Setup: given a security parameter λ, $\mathcal{G}$ runs the algorithm Setup to provide public parameters $parm$ to $\mathcal{A}_3$, while keeps the master secret key msk for itself.

2 Phase 1: $\mathcal{A}_3$ can make $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle$, $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle$, $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle$, $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle$ and $\mathcal{O}_{Dec}$ query

$\langle C_{ID,t}, ID, t\rangle$ adaptively in this phase. The constraints are that $\langle ID^*\rangle$ does not appear in any $\mathcal{O}_{ExtKey}$ query, $\langle ID^*, t^*\rangle$ does not appear in any $\mathcal{O}_{TestKey}$ query, $\mathcal{O}_{RevKey}$ query and $\mathcal{O}_{EvoKey}$ query.

3 Challenge: $\mathcal{A}_3$ outputs two equal length messages m_0 and m_1, $\mathcal{G}$ randomly chooses $b \in \{0,1\}$ and encrypts the message m_b via algorithm Encrypt to output a challenge ciphertext C_{ID^*,t^*}. Finally, return the challenge ciphertext C_{ID^*,t^*} to $\mathcal{A}_3$.

4 Phase 2: $\mathcal{A}_3$ continues to issue more queries as below:

- $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle \neq \langle ID^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle \neq \langle ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle \neq \langle ID^*, t^*\rangle\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle \neq \langle ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.
- $\mathcal{O}_{Dec}$ query $\langle C_{ID,t}, ID, t\rangle \neq \langle C_{ID^*,t^*}, ID^*, t^*\rangle$: $\mathcal{G}$ performs as in Phase 1.

5 Guess: $\mathcal{A}_3$ outputs a guess $b' \in \{0,1\}$ and wins if $b = b'$.

We define the advantage of $\mathcal{A}_3$ in the above game as

$$\mathbf{Adv}^{\text{IND-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_3}(\lambda) = |\Pr[b = b'] - \frac{1}{2}|.$$

Definition 3.4: We say that a R-IBEET scheme is IND-sID-CCA secure, if for each type-III adversary, $\mathbf{Adv}^{\text{IND-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_3}(\lambda)$ is negligible in the security parameter λ.

4 Construction

We present the construction of R-IBEET below.

1 Setup(λ): on input a security parameter $\lambda \in \mathbb{Z}^+$, this algorithm outputs the public parameter $parm$ and the master secret key msk as below:

a Generate a group $\mathbb{G}$ of prime order p and randomly selects a generator g of $\mathbb{G}$.

b Select seven hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p$, $H_2 : \{0,1\}^* \to \mathbb{Z}_p$, $H_3 : \{0,1\}^\lambda \to \mathbb{Z}_p$, $H_4 : \mathbb{G} \to \{0,1\}^{\lambda+l}$, $H_5 : \mathbb{G} \to \{0,1\}^{\lambda+l}$, $H_6 : \mathbb{G} \to \{0,1\}^{4l}$, $H_7 : \mathbb{G} \to \mathbb{G}$, where l is bit-length of elements in $\mathbb{Z}_p$.

c Randomly pick $(s, s') \in \mathbb{Z}_p^2$ as the master secret key msk.

d Pick $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6 \in \{0,1\}^\lambda$ randomly.

e Output $parm = \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, and $msk = (s, s')$.

2 Extract(msk, ID): given $ID \in \{0,1\}^*$, the algorithm computes the private key $SK_{ID} = H_1(s||ID)$ of the user and the corresponding public key $PK_{ID} = g^{SK_{ID}}$.

3 TimeKeyTest(SK_{ID}, t): the algorithm computes test key $TestKey_{ID,t} = H_2(SK_{ID}||t)$ as the test key of the tester and the corresponding public test key $PK_{test,ID,t} = g^{TestKey_{ID,t}}$, where t is the time tag.

4 TimeKeyRev(msk, ID, t): the algorithm computes revoke key $RevKey_{ID,t} = H_1(s'||ID||t)$ as the revoke key of the user and the corresponding public revoke key $PK_{rev,ID,t} = g^{RevKey_{ID,t}}$, where t is the time tag.

5 TimeKeyEvo($msk, ID, t, TestKey_{ID,t}$): the algorithm first computes $RevKey_{ID,t} = H_1(s'||ID||t)$, then sets $EvoKey_{ID,t} = (RevKey_{ID,t}, TestKey_{ID,t})$ as the evolve key of the cloud server.

6 Encrypt($m, PK_{ID}, PK_{rev,ID,t}, PK_{test,ID,t}$): to encrypt a message $m \in \{0,1\}^\lambda$, the algorithm generates a ciphertext $C_{ID,t} = (C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}, C^5_{ID,t})$ as follows:

a Compute three points:

$$P_1 = (H_3(\gamma_1 \oplus m), H_3(\gamma_2 \oplus m)),$$
$$P_2 = (H_3(\gamma_3 \oplus m), H_3(\gamma_4 \oplus m)),$$
$$P_3 = (H_3(\gamma_5 \oplus m), H_3(\gamma_6 \oplus m)).$$

b Construct a quadratic polynomial $f(x)$ that passes through three points: P_1, P_2, P_3.

c Compute two random points $(x_1, y_1), (x_2, y_2)$ on $f(x)$, where $x_1, x_2 \in \{0,1\}^l$.

d Pick two random values $r_1, r_2 \in \mathbb{Z}_p$, then compute

$$C^1_{ID,t} = g^{r_1}, C^2_{ID,t} = g^{r_2},$$
$$C^3_{ID,t} = (m||r_2) \oplus H_4(PK^{r_1}_{ID}) \oplus H_5(PK^{r_2}_{Rev,ID,t}),$$
$$C^4_{ID,t} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID,t}),$$
$$C^5_{ID,t} = H_7(H_5(PK^{r_2}_{Rev,ID,t}), C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}).$$

7 Decrypt($C_{ID,t}, SK_{ID}, RevKey_{ID,t}$): let $C_{ID,t} = (C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}, C^5_{ID,t})$. This algorithm performs the following steps:

a Compute $TestKey_{ID,t} = H_2(SK_{ID}||t)$.

b Recover $m||r_2$ by computing

$$C^3_{ID,t} \oplus H_4((C^1_{ID,t})^{SK_{ID}}) \oplus H_5((C^2_{ID,t})^{RevKey_{ID,t}}).$$

c Recover $x_1||x_2||y_1||y_2$ by computing

$$C^4_{ID,t} \oplus H_6((C^2_{ID,t})^{TestKey_{ID,t}}.$$

d Compute P_1, P_2, P_3 with m and the six random numbers γ_1, γ_2, γ_3, γ_4, γ_5, γ_6, then reconstruct a quadratic polynomial $f(x)$ with them. If $C^2_{ID,t} = g^{r_2}$, $C^5_{ID,t} = H_7((C^2_{ID,t})^{RevKey_{ID,t}}, C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t})$, $f(x_1) = y_1$, $f(x_2) = y_2$, output m; or an error symbol $\perp$ otherwise.

8 Test($C_{ID_i,t_\alpha}, TestKey_{ID_i,t_\alpha}, C_{ID_j,t_\beta}, TestKey_{ID_j,t_\beta}$): the algorithm computes

$$\begin{aligned} x_1||x_2||y_1||y_2 &= C^4_{ID_i,t_\alpha} \\ &\quad \oplus H_6((C^2_{ID_i,t_\alpha})^{TestKey_{ID_i,t_\alpha}}), \\ x'_1||x'_2||y'_1||y'_2 &= C^4_{ID_j,t_\beta} \\ &\quad \oplus H_6((C^2_{ID_j,t_\beta})^{TestKey_{ID_j,t_\beta}}). \end{aligned}$$

Then, it reconstructs quadratic polynomials

$$\begin{aligned} f(x) &\leftarrow ((x_1, y_1), (x_2, y_2), (x'_1, y'_1)), \\ f(x)' &\leftarrow ((x'_1, y'_1), (x'_2, y'_2), (x_1, y_1)), \end{aligned}$$

and retrieves secrets ϑ and ϑ' hidden within those two quadratic polynomials

$$\vartheta = f(0), \vartheta' = f(0)'.$$

It outputs 1 if $\vartheta = \vartheta'$ holds, which indicates $m = m'$, or 0 otherwise.

9 Revoke(ID, t): if a user with identity ID needs to be revoked at time t, the PKG stops issuing the revoke key $RevKey_{ID,t}$ for the user at time t.

10 CiphertextEvolve($C_{ID,t}, ID, EvoKey_{ID,t}, EvoKey_{ID,t'}$): to transform a ciphertext $C_{ID,t}$ of ID at time t into a new ciphertext $C_{ID,t'}$ of ID at the current time $t' > t$, the algorithm computes:

$$\begin{aligned} C^1_{ID,t'} &= C^1_{ID,t}, C^2_{ID,t'} = C^2_{ID,t}, \\ C^3_{ID,t'} &= C^3_{ID,t} \oplus H_5((C^2_{ID,t})^{RevKey_{ID,t}}) \\ &\quad \oplus H_5((C^2_{ID,t})^{RevKey_{ID,t'}}), \\ C^4_{ID,t'} &= C^4_{ID,t} \oplus H_6((C^2_{ID,t})^{TestKey_{ID,t}}) \\ &\quad \oplus H_6((C^2_{ID,t})^{TestKey_{ID,t'}}), \\ C^5_{ID,t'} &= H_7(H_5(C^2_{ID,t})^{RevKey_{ID,t'}}, C^1_{ID,t'}, \\ &\quad C^2_{ID,t'}, C^3_{ID,t'}, C^4_{ID,t'}). \end{aligned}$$

Then, it outputs a ciphertext

$$C_{ID,t'} = (C^1_{ID,t'}, C^2_{ID,t'}, C^3_{ID,t'}, C^4_{ID,t'}, C^5_{ID,t'}).$$

5 Security analysis

Below we will formally give security proofs of our scheme via a sequence of games.

Theorem 1: The R-IBEET construction is OW-sID-CCA secure against any probabilistic polynomial time (PPT) type-I adversaries provided that CDH assumption holds.

Proof: Suppose $\mathcal{A}_1$ is a PPT type-I adversary who manages to break the OW-sID-CCA security with advantage $\mathbf{Adv}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}(\lambda)$. Assume it makes queries to $H_d(d = 1, 2, 3, 4, 5, 6, 7)$ oracles for up to q_{H_d} times, and it makes at most q_{ExtKey} private key queries, $q_{TestKey}$ test key queries, q_{RevKey} revoke key queries, q_{EvoKey} evolve key queries, and q_{Dec} decryption queries. We analyse the security via a series of games as below.

Game 1.0

Init: $\mathcal{A}_1$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}^2_p$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the following way.

- $\mathcal{O}_{H_1}$ query $\langle \mu_1, \mu_2 \rangle$: on input μ_1 and μ_2, the challenger randomly picks $h_1 \in \mathbb{Z}_p$ and sends h_1 to $\mathcal{A}_1$. The challenger adds (μ_1, μ_2, h_1) into table T_1 for $\mathcal{O}_{H_1}$.
- $\mathcal{O}_{H_2}$ query $\langle \mu_1, \mu_2 \rangle$: on input μ_1 and μ_2, the challenger randomly picks $h_2 \in \mathbb{Z}_p$ and sends h_2 to $\mathcal{A}_1$. The challenger adds (μ_1, μ_2, h_2) into table T_2 for $\mathcal{O}_{H_2}$.
- $\mathcal{O}_{H_3}$ query $\langle \mu_1 \rangle$: on input μ_1, the challenger randomly picks $h_3 \in \mathbb{Z}_p$ and sends h_3 to $\mathcal{A}_1$. The challenger adds (μ_1, h_3) into table T_3 for $\mathcal{O}_{H_4}$.
- $\mathcal{O}_{H_4}$ query $\langle \mu_1 \rangle$: on input μ_1, the challenger randomly picks $h_4 \in \{0, 1\}^{\lambda+l}$ and sends h_4 to $\mathcal{A}_1$. The challenger adds (μ_1, h_4) into table T_4 for $\mathcal{O}_{H_4}$.
- $\mathcal{O}_{H_5}$ query $\langle \mu_1 \rangle$: on input μ_1, the challenger randomly picks $h_5 \in \{0, 1\}^{\lambda+l}$ and sends h_5 to $\mathcal{A}_1$. The challenger adds (μ_1, h_5) into table T_5 for $\mathcal{O}_{H_5}$.
- $\mathcal{O}_{H_6}$ query $\langle \mu_1 \rangle$: on input μ_1, the challenger randomly picks $h_6 \in \{0, 1\}^{4l}$ and sends h_6 to $\mathcal{A}_1$. The challenger adds (μ_1, h_6) into table T_6 for $\mathcal{O}_{H_6}$.
- $\mathcal{O}_{H_7}$ query $\langle \mu_1, \mu_2, \mu_3, \mu_4, \mu_5 \rangle$: on input $\mu_1, \mu_2, \mu_3, \mu_4$ and μ_5, the challenger randomly

picks $h_7 \in \mathbb{G}$ sends h_7 to $\mathcal{A}_1$. The challenger adds $(\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, h_7)$ into table T_7 for $\mathcal{O}_{H_7}$.

- $\mathcal{O}_{ExtKey}$ query $\langle ID \rangle$: on input an identity ID, the challenger sends $(SK_{ID}, PK_{ID}) \leftarrow \mathsf{Extract}(parm, msk, ID)$ to $\mathcal{A}_1$.
- $\mathcal{O}_{TestKey}$ query $\langle ID, t \rangle$: on input an identity ID and a time tag t, the challenger sends $(TestKey_{ID,t}, PK_{test,ID,t}) \leftarrow \mathsf{TimeKeyTest}(SK_{ID}, t)$ to $\mathcal{A}_1$.
- $\mathcal{O}_{RevKey}$ query $ID, t\rangle$: on input an identity ID and a time tag t, the challenger sends $(RevKey_{ID,t}, PK_{rev,ID,t}) \leftarrow \mathsf{TimeKeyRev}(msk, ID, t)$ to $\mathcal{A}_1$.
- $\mathcal{O}_{EvoKey}$ query $\langle ID, t \rangle$: on input an identity ID and a time tag t, the challenger sends $EvoKey_{ID,t} \leftarrow \mathsf{TimeKeyEvo}(msk, ID, t, TestKey_{ID,t})$ to $\mathcal{A}_1$.
- $\mathcal{O}_{Dec}$ query $\langle C_{ID,t}, ID, t \rangle$: on input a ciphertext $C_{ID,t}$, an identity ID and a time tag t, the challenger sends $m \leftarrow \mathsf{Decrypt}(C, SK_{ID}, RevKey_{ID,t})$ to $\mathcal{A}_1$.

3 $m \leftarrow \{0,1\}^\lambda$, $r_1, r_2 \leftarrow \mathbb{Z}_p$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$
\begin{aligned}
&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2},\\
&C^3_{ID^*,t^*} = (m||r_2) \oplus H_4(PK^{r_1}_{ID^*}) \oplus H_5(PK^{r_2}_{Rev,ID^*,t^*}),\\
&C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),\\
&C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*},\\
&\qquad C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).
\end{aligned}
$$

The points (x_1, y_1) and (x_2, y_2) are generated randomly on a quadratic polynomial $f(x)$ which is generated by passing through three points:
$P_1 = (H_3(\gamma_1 \oplus m), H_3(\gamma_2 \oplus m))$,
$P_2 = (H_3(\gamma_3 \oplus m), H_3(\gamma_4 \oplus m))$,
$P_3 = (H_3(\gamma_5 \oplus m), H_3(\gamma_6 \oplus m))$.

4 $m' \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(state, C_{ID^*,t^*})$.

Denote by $\mathbf{S}_{1.0}$ the event that $m = m'$ in Game 1.0. Then

$$\mathbf{Adv}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}(\lambda) = \Pr[\mathbf{S}_{1.0}]. \tag{1}$$

Game 1.1

Init: $\mathcal{A}_1$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}_p^2$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the following way:

- $\mathcal{O}_{H_1}$ query $\langle \mu_1, \mu_2 \rangle$, $\mathcal{O}_{H_2}$ query $\langle \mu_1, \mu_2 \rangle$, $\mathcal{O}_{H_3}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_4}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_5}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_6}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_7}$ query $\langle \mu_1, \mu_2, \mu_3, \mu_4, \mu_5 \rangle$, $\mathcal{O}_{ExtKey}$ query $\langle ID \rangle$, $\mathcal{O}_{TestKey}$ query $\langle ID, t \rangle$, $\mathcal{O}_{RevKey}$ query $\langle ID, t \rangle$, $\mathcal{O}_{EvoKey}$ query $\langle ID, t \rangle$: Same as those in Game 1.0.
- $\mathcal{O}_{Dec}$ query $\langle C_{ID,t}, ID, t \rangle$: The challenger queries to H_4 on input $(C^1_{ID,t})^{SK_{ID}}$ to get h_4 and queries to H_5 on input $(C^2_{ID,t})^{RevKey_{ID,t}}$ to get h_5. It computes $m'||r'_2 = C^3_{ID,t} \oplus h_4 \oplus h_5$. It then computes P_1, P_2 and P_3 with m' and the six random numbers $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6$, and reconstructs a quadratic polynomial $f(x)$ with them. Then the challenger queries $(H_5((C^2_{ID,t})^{RevKey_{ID,t}}, C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t})$ to $\mathcal{O}_{H_7}$ to get answer h_7. If either of $C^2_{ID,t} = g^{r_2}$, $C^5_{ID,t} = h_7$, $f(x_1) = y_1$, $f(x_2) = y_2$ fails to hold, return $\perp$ to $\mathcal{A}_1$; Otherwise, return m' to $\mathcal{A}_1$.

3 $m \leftarrow \{0,1\}^\lambda$, $r_1, r_2 \leftarrow \mathbb{Z}_p$, $W^*_{1.1} \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$
\begin{aligned}
&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2},\\
&C^3_{ID^*,t^*} = (m||r_2) \oplus H_4(PK^{r_1}_{ID^*}) \oplus W^*_{1.1},\\
&C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),\\
&C^5_{ID^*,t^*} = H_7(W^*_{1.1}, C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*},\\
&\qquad C^4_{ID^*,t^*}).
\end{aligned}
$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0. Add the tuple $((C^2_{ID^*,t^*})^{RevKey_{ID^*,t^*}}, W^*_{1.1})$ into table T_5 for $\mathcal{O}_{H_5}$, and tuple $(W^*_{1.1}, C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ into table T_7 for $\mathcal{O}_{H_7}$.

4 $m' \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(state, C_{ID^*,t^*})$.

Let $\mathbf{S}_{1.1}$ be the event that $m = m'$ in Game 1.1. It is because the random oracle is ideal that there is no difference between Games 1.1 and 1.0. Then

$$\Pr[\mathbf{S}_{1.0}] = \Pr[\mathbf{S}_{1.1}]. \tag{2}$$

Game 1.2

Init: $\mathcal{A}_1$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}_p^2$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the same way as those in Game 1.1.

3 $r_1, r_2 \leftarrow \mathbb{Z}_p$, $W_{2.2}^* \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, C_{ID^*,t^*}^3, C_{ID^*,t^*}^4, C_{ID^*,t^*}^5)$ defined as follows:

$$
\begin{aligned}
&C_{ID^*,t^*}^1 = g^{r_1}, C_{ID^*,t^*}^2 = g^{r_2},\\
&C_{ID^*,t^*}^3 = W_{2.2}^*,\\
&C_{ID^*,t^*}^4 = (x_1||x_2||y_1||y_2) \oplus H_6(PK_{test,ID^*,t^*}^{r_2}),\\
&C_{ID^*,t^*}^5 = H_7(W_{2.1}^*, C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, C_{ID^*,t^*}^3, C_{ID^*,t^*}^4).
\end{aligned}
$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0. Add the tuple $((C_{ID^*,t^*}^2)^{RevKey_{ID^*,t^*}}, W_{2.2}^* \oplus H_4(C_{ID^*,t^*}^1)^{SK_{ID^*}} \oplus (m||r_2))$ into table T_5 for $\mathcal{O}_{H_5}$, and tuple $(W_{2.2}^* \oplus H_4(C_{ID^*,t^*}^1)^{SK_{ID^*}} \oplus (m||r_2), C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, C_{ID^*,t^*}^3, C_{ID^*,t^*}^4, C_{ID^*,t^*}^5)$ into table T_7 for $\mathcal{O}_{H_7}$.

4 $m' \leftarrow \mathcal{A}_1^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(state, C_{ID^*,t^*})$.

- If $\mathcal{A}_1$ queries $(C_{ID^*,t^*}^2)^{RevKey_{ID^*,t^*}}$ to $\mathcal{O}_{H_5}$, then denote this event by $\mathbf{E}_{1.1}$.
- If $\mathcal{A}_1$ queries $(C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, (C_{ID^*,t^*}^3)', C_{ID^*,t^*}^4, C_{ID^*,t^*}^5)$ to $\mathcal{O}_{Dec}$, where $(C_{ID^*,t^*}^3)' \neq (C_{ID^*,t^*}^3)$, $\perp$ is returned.

Let $\mathbf{S}_{1.2}$ be the event that $m = m'$ in Game 1.2. It is because the random oracle is ideal that the challenge ciphertexts in Game 1.1 and Game 1.2 are same in distribution. Hence, there will be no difference between Game 1.2 and $Game1.1$ if $\mathbf{E}_{1.1}$ does not happen. Then

$$|\Pr[\mathbf{S}_{1.1}] - \Pr[\mathbf{S}_{1.2}]| \leq \Pr[\mathbf{E}_{1.1}]. \tag{3}$$

In the following lemma, we will prove that the probability that $\mathbf{E}_{1.1}$ happens is negligible.

Lemma 1: The probability that the event $\mathbf{E}_{1.1}$ happens in $Game1.2$ is negligible provided that CDH assumption holds.

Proof: Assume that the probability that $\mathbf{E}_{1.1}$ happens is non-negligible. We can build a PPT algorithm $\mathcal{B}_1$ who invokes $\mathcal{A}_1'$ with the aim to solve the CDH problem. Given an instance of the CDH problem: $(\mathbb{G}, p, g, g^a, g^c)$, $\mathcal{B}_1$ works in the following way:

1 $\mathcal{B}_1$ sets $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$. It selects $(s, s') \leftarrow \mathbb{Z}_p^2$, and sets $msk = (s, s')$, $PK_{rev,ID^*,t^*} = g^a$, which implies $RevKey_{ID^*,t^*} = a$ and $EvoKey_{ID^*,t^*} = (a, H_2(SK_{ID^*}||t^*))$. Then it runs algorithm TimeKeyRev to generate all other revoke key $RevKey_{ID,t}$ and the corresponding public revoke key $PK_{rev,ID,t}$, where $(ID, t) \neq (ID^*, t^*)$.

2 $state \leftarrow \mathcal{A}_1'^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the following way.

- $\mathcal{O}_{H_1}$ query $\langle \mu_1, \mu_2 \rangle$, $\mathcal{O}_{H_2}$ query $\langle \mu_1, \mu_2 \rangle$, $\mathcal{O}_{H_3}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_4}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_5}$ query $\langle \mu_1 \rangle$ $\mathcal{O}_{H_6}$ query $\langle \mu_1 \rangle$, $\mathcal{O}_{H_7}$ query $\langle \mu_1, \mu_2, \mu_3, \mu_4, \mu_5 \rangle$, $\mathcal{O}_{ExtKey}$ query $\langle ID \rangle$, $\mathcal{O}_{TestKey}$ query $\langle ID, t \rangle$, $\mathcal{O}_{RevKey}$ query $\langle ID, t \rangle$, $\mathcal{O}_{EvoKey}$ query $\langle ID, t \rangle$: Same as those in Game 1.2.
- $\mathcal{O}_{Dec}$ query $\langle C, ID, t \rangle$: $\mathcal{B}_1$ searches table T_4 on input $(C_{ID,t}^1)^{SK_{ID}}$ to get h_4, searches table T_5 to get h_5, searches table T_6 on input $(C_{ID,t}^2)^{TestKey_{ID,t}}$ to get h_6 and searches T_7 on input $(h_5, C_{ID,t}^1, C_{ID,t}^2, C_{ID,t}^3, C_{ID,t}^4)$ to get h_7, then it computes $x_1||x_2||y_1||y_2 = C_{ID,t}^4 \oplus h_6$. For each tuple (μ_1, h_5), $\mathcal{B}_1$ performs as follows:
  - a Compute $m'||r_2' = C_{ID,t}^3 \oplus h_4 \oplus h_5$.
  - b Compute P_1, P_2, P_3 with m' and the six random numbers $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6$, then reconstruct a quadratic polynomial $f(x)$ with them.
  - c If $C_{ID,t}^2 = g^{r_2}$, $C_{ID,t}^5 = H_7(h_5, C_{ID,t}^1, C_{ID,t}^2, C_{ID,t}^3, C_{ID,t}^4)$, $f(x_1) = y_1$ and $f(x_2) = y_2$ all hold, return m' to $\mathcal{A}_1'$; if there exists no such tuple in table T_5, return $\perp$ to $\mathcal{A}_1'$.

3 $r_1 \leftarrow \mathbb{Z}_p$, $(W_{2.2}^*)' \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, C_{ID^*,t^*}^3, C_{ID^*,t^*}^4, C_{ID^*,t^*}^5)$ defined as follows:

$$
\begin{aligned}
&C_{ID^*,t^*}^1 = g^{r_1}, C_{ID^*,t^*}^2 = g^c,\\
&C_{ID^*,t^*}^3 = (W_{2.2}^*)',\\
&C_{ID^*,t^*}^4 = (x_1||x_2||y_1||y_2) \oplus H_6((C_{ID,t}^2)^{TestKey_{ID,t}}),\\
&C_{ID^*,t^*}^5 = H_7((W_{2.1}^*)', C_{ID^*,t^*}^1, C_{ID^*,t^*}^2, C_{ID^*,t^*}^3, C_{ID^*,t^*}^4).
\end{aligned}
$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0.

4 $m' \leftarrow \mathcal{A}_1'^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey},\mathcal{O}_{Dec}}(state, C_{ID^*,t^*})$.

- If $\mathcal{A}_1'$ queries $(C_{ID^*,t^*}^2)^{RevKey_{ID^*,t^*}} = g^{ac}$ to $\mathcal{O}_{H_5}$, then denote this event by $\mathbf{D}_{1.1}$.

- If $\mathcal{A}_1'$ queries $(C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, (C^3_{ID^*,t^*})', C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ to $\mathcal{O}_{Dec}$, where $(C^3_{ID^*,t^*})' \neq (C^3_{ID^*,t^*})$, $\perp$ is returned.

It is obvious that both of randomness and correctness properties hold for the above simulation. Hence, we just need to analyse the decryption queries as below:

1 In the case where $\mathcal{A}_1'$ has already queried $(C^2_{ID,t})^a$ to H_5 before a decryption query $(C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}, C^5_{ID,t})$, the value h_5 can be uniquely determined. Hence, the simulation of the decryption oracle is perfect.

2 In the case where $\mathcal{A}_1'$ has not yet queried $(C^2_{ID,t})^a$ to H_5 before a decryption query $(C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}, C^5_{ID,t})$, $\perp$ is returned. The simulation fails if $(C^1_{ID,t}, C^2_{ID,t}, C^3_{ID,t}, C^4_{ID,t}, C^5_{ID,t})$ is a valid ciphertext. However, it is because the random oracle is ideal that the probability that it occurs is no more than $\frac{1}{2^{\lambda+l}}$.

Denote the event $\mathbf{D}_{1.2}$ that a valid ciphertext is rejected in the simulation. Then we have $\Pr[\mathbf{D}_{1.2}] \leq \frac{q_{Dec}}{2^{\lambda+l}}$, which is negligible. Thus, $\mathcal{B}_1$ performs decryption simulation correctly except with negligible probability.

Probability of successful simulation: since there is no abort in the simulated game, the probability of successful simulation is 1.

Analysis: the probability that $(C^2_{ID^*,t^*})^a$ is queried to $\mathcal{O}_{H_5}$ by $\mathcal{A}_1'$ is $\Pr[\mathbf{D}_{1.1}]$. Hence, $\mathcal{B}_1$ can break the CDH assumption with probability $\frac{\Pr[\mathbf{D}_{1.1}]}{q_{H_5}}$. That is, $\Pr[\mathbf{D}_{1.1}] = q_{H_5}\mathbf{Adv}^{\text{CDH}}$. Furthermore, if $\mathbf{D}_{1.2}$ does not occur, the simulated game is indistinguishable from the Game 1.2. Hence, $\Pr[\mathbf{D}_{1.1}|\neg\mathbf{D}_{1.2}] = \Pr[\mathbf{E}_{1.1}]$.

$$\begin{aligned}\Pr[\mathbf{D}_{1.1}] &= Pr[\mathbf{D}_{1.1}|\mathbf{D}_{1.2}]Pr[\mathbf{D}_{1.2}] \\ &\quad + Pr[\mathbf{D}_{1.1}|\neg\mathbf{D}_{1.2}]Pr[\neg\mathbf{D}_{1.2}] \\ &\geq Pr[\mathbf{D}_{1.1}|\neg\mathbf{D}_{1.2}]Pr[\neg\mathbf{D}_{1.2}] \\ &= Pr[\mathbf{E}_{1.1}](1 - Pr[\mathbf{D}_{1.2}]) \\ &\geq Pr[\mathbf{E}_{1.1}] - Pr[\mathbf{D}_{1.2}].\end{aligned}$$

Hence, $q_{H_5}\mathbf{Adv}^{\text{CDH}} \geq \Pr[\mathbf{E}_{1.1}] - \frac{q_{Dec}}{2^{\lambda+l}}$. That is,

$$\Pr[\mathbf{E}_{1.1}] \leq q_{H_5}\mathbf{Adv}^{\text{CDH}} + \frac{q_{Dec}}{2^{\lambda+l}},$$

which is negligible. This completes the proof of Lemma 1. □

Finally, we analyse the challenge ciphertext C_{ID^*,t^*} in Game 1.2:

$$\begin{aligned}&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^c, \\ &C^3_{ID^*,t^*} = W^*_{2.2}, \\ &C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}), \\ &C^5_{ID^*,t^*} = H_7(W^*_{2.1}, C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).\end{aligned}$$

It is because the random oracle is ideal and the hash function is one-way that $\mathcal{A}_1$ can figure out m by the challenge ciphertext with negligible probability ϵ. Then,

$$\Pr[\mathbf{S}_{1.2}] \leq \epsilon. \tag{4}$$

Combining equations (1)–(4), we have that

$$\mathbf{Adv}^{\text{OW-sID-CCA}}_{\text{R-IBEET},\mathcal{A}_1}(\lambda) \leq q_{H_5}\mathbf{Adv}^{\text{CDH}} + \frac{q_{Dec}}{2^{\lambda+l}} + \epsilon,$$

which is negligible. The proof of Theorem 1 is complete. □

Theorem 2: The R-IBEET construction is OW-sID-CPA secure against any PPT type-II adversaries provided that CDH assumption holds.

Proof: Suppose $\mathcal{A}_2$ is a PPT type-II adversary who manages to break the OW-sID-CPA security with advantage $\mathbf{Adv}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}(\lambda)$. Assume it makes queries to $H_d(d = 1, 2, 3, 4, 5, 6, 7)$ oracles for up to q_{H_d} times, and it makes at most q_{ExtKey} private key queries, $q_{TestKey}$ test key queries, q_{RevKey} revoke key queries, and q_{EvoKey} evolve key queries. We analyse the security via a series of games as below.

Game 2.0

Init: $\mathcal{A}_2$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}_p^2$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the following way.

- $\mathcal{O}_{H_1}$ query $\langle\mu_1, \mu_2\rangle$, $\mathcal{O}_{H_2}$ query $\langle\mu_1, \mu_2\rangle$, $\mathcal{O}_{H_3}$ query $\langle\mu_1\rangle$, $\mathcal{O}_{H_4}$ query $\langle\mu_1\rangle$, $\mathcal{O}_{H_5}$ query $\langle\mu_1\rangle$, $\mathcal{O}_{H_6}$ query $\langle\mu_1\rangle$, $\mathcal{O}_{H_7}$ query $\langle\mu_1, \mu_2, \mu_3, \mu_4, \mu_5\rangle$, $\mathcal{O}_{ExtKey}$ query $\langle ID\rangle$, $\mathcal{O}_{TestKey}$ query $\langle ID, t\rangle$, $\mathcal{O}_{RevKey}$ query $\langle ID, t\rangle$, $\mathcal{O}_{EvoKey}$ query $\langle ID, t\rangle$: Same as those in Game 1.0.

3 $m \leftarrow \{0,1\}^\lambda$, $r_1, r_2 \leftarrow \mathbb{Z}_p$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$\begin{aligned}&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2}, \\ &C^3_{ID^*,t^*} = (m||r_2) \oplus H_4(PK^{r_1}_{ID^*}) \\ &\qquad \oplus H_5(PK^{r_2}_{Rev,ID^*,t^*}), \\ &C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}), \\ &C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, \\ &\qquad C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).\end{aligned}$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0.

4 $m' \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}(state, C_{ID^*,t^*})$.

Denote by $\mathbf{S}_{2.0}$ the event that $m = m'$ in Game 2.0. Then

$$\mathbf{Adv}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}(\lambda) = \Pr[\mathbf{S}_{2.0}]. \tag{5}$$

Game 2.1

Init: $\mathcal{A}_2$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}_p^2$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}$ $(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the same way as those in Game 2.0.

3 $m \leftarrow \{0,1\}^\lambda$, $r_1, r_2 \leftarrow \mathbb{Z}_p$, $W^*_{1.1} \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$\begin{aligned}
&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2},\\
&C^3_{ID^*,t^*} = (m||r_2) \oplus W^*_{1.1} \oplus H_5(PK^{r_2}_{Rev,ID^*,t^*}),\\
&C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),\\
&C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).
\end{aligned}$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0. Add the tuple $((C^1_{ID^*,t^*})^{RevKey_{ID^*,t^*}}, W^*_{1.1})$ into table T_4 for $\mathcal{O}_{H_4}$.

4 $m' \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}(state, C_{ID^*,t^*})$.

Let $\mathbf{S}_{2.1}$ be the event that $m = m'$ in Game 2.1. It is because the random oracle is ideal that there is no difference between Game 2.1 and Game 2.0. Then

$$\Pr[\mathbf{S}_{2.0}] = \Pr[\mathbf{S}_{2.1}]. \tag{6}$$

Game 2.2

Init: $\mathcal{A}_2$ outputs an identity ID^* and a time tag t^* to be challenged.

1 $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$, $(s, s') \leftarrow \mathbb{Z}_p^2$, $msk = (s, s')$.

2 $state \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}$ $(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the same way as those in Game 2.1.

3 $m \leftarrow \{0,1\}^\lambda$, $r_1, r_2 \leftarrow \mathbb{Z}_p$, $W^*_{2.1} \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$\begin{aligned}
&C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2},\\
&C^3_{ID^*,t^*} = (m||r_2) \oplus W^*_{2.1},\\
&C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),\\
&C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).
\end{aligned}$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0. Add the tuple $((C^2_{ID^*,t^*})^{RevKey_{ID^*,t^*}}, W^*_{2.1} \oplus H_5((C^2_{ID^*,t^*})^{RevKey_{ID^*,t^*}}))$ into table T_4 for $\mathcal{O}_{H_4}$.

4 $m' \leftarrow \mathcal{A}_2^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}(state, C_{ID^*,t^*})$.

- If $\mathcal{A}_2$ queries $(C^1_{ID^*,t^*})^{SK_{ID^*}}$ to $\mathcal{O}_{H_4}$, then denote this event $\mathbf{E}_{2.1}$.

Let $\mathbf{S}_{2.2}$ be the event that that $m = m'$ in Game 2.2. It is because the random oracle is ideal that the challenge ciphertexts in Game 2.1 and Game 2.2 are same in distribution. Hence, there will be no difference between Game 2.2 and Game 2.1 if $\mathbf{E}_{2.1}$ does not happen. Then

$$|\Pr[\mathbf{S}_{2.1}] - \Pr[\mathbf{S}_{2.0}]| \leq \Pr[\mathbf{E}_{2.1}]. \tag{7}$$

In the following lemma, we will prove that the probability that $\mathbf{E}_{2.1}$ happens is negligible.

Lemma 2: The probability that the event $\mathbf{E}_{2.1}$ happens in Game 2.2 is negligible provided that CDH assumption holds.

Proof: Assume that the probability that $\mathbf{E}_{2.1}$ happens is non-negligible. We can build a PPT algorithm $\mathcal{B}_2$ who invokes $\mathcal{A}'_2$ with the aim to solve the CDH problem. Given an instance of the CDH problem: $(\mathbb{G}, p, g, g^a, g^c)$, $\mathcal{B}_2$ works in the following way.

1 $\mathcal{B}_2$ sets $parm \leftarrow \{\mathbb{G}, p, g, H_1, H_2, H_3, H_4, H_5, H_6, H_7, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6\}$. It selects $(s, s') \leftarrow \mathbb{Z}_p^2$, and sets $msk = (s, s')$, $PK_{ID^*} = g^a$, which implies $SK_{ID^*} = a$ and $EvoKey_{ID^*,t^*} = (H_1(s'||ID^*||t^*), H_2(a||t^*))$.

2 $state \leftarrow \mathcal{A}'^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}_2$ $(parm, ID \in \{ID_1, \ldots, ID_n\}, t)$, where the oracles work in the same way as those in Game 2.2.

3 $m \leftarrow \{0,1\}^\lambda$, $r_2 \leftarrow \mathbb{Z}_p$, $(W^*_{2.1})' \leftarrow \{0,1\}^{\lambda+l}$, $C_{ID^*,t^*} = (C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}, C^5_{ID^*,t^*})$ defined as follows:

$$C^1_{ID^*,t^*} = g^c, C^2_{ID^*,t^*} = g^{r_2},$$

$$C^3_{ID^*,t^*} = (m||r_2) \oplus (W^*_{2.1})',$$
$$C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),$$
$$C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).$$

The points (x_1, y_1) and (x_2, y_2) are generated in the same way as that in Game 1.0.

4 $m' \leftarrow \mathcal{A}'^{\mathcal{O}_{H_1},\mathcal{O}_{H_2},\mathcal{O}_{H_3},\mathcal{O}_{H_4},\mathcal{O}_{H_5},\mathcal{O}_{H_6},\mathcal{O}_{H_7},\mathcal{O}_{ExtKey},\mathcal{O}_{TestKey},\mathcal{O}_{RevKey},\mathcal{O}_{EvoKey}}_2(state, C_{ID^*,t^*})$.

- If $\mathcal{A}'_2$ queries $(C^1_{ID^*,t^*})^{SK_{ID^*}} = g^{ac}$ to $\mathcal{O}_{H_4}$, then denote this event by $\mathbf{D}_{2.1}$.

Probability of successful simulation: since there is no abort in the simulated game, the probability of successful simulation is 1.

Analysis: the probability that $(C^1_{ID^*,t^*})^a$ is queried to $\mathcal{O}_{H_4}$ by $\mathcal{A}'_2$ is $\Pr[\mathbf{D}_{2.1}]$. Hence, $\mathcal{B}_2$ can break the CDH assumption with probability $\frac{\Pr[\mathbf{D}_{2.1}]}{q_{H_4}}$. That is, $\Pr[\mathbf{D}_{2.1}] = q_{H_4}\mathbf{Adv}^{\text{CDH}}$. Furthermore, the simulated game is indistinguishable from the Game 2.2. Hence, $\Pr[\mathbf{D}_{2.1}] = \Pr[\mathbf{E}_{2.1}]$.

Hence, $q_{H_4}\mathbf{Adv}^{\text{CDH}} = \Pr[\mathbf{E}_{2.1}]$, which is negligible. This completes the proof of Lemma 2. □

Finally, we analyse the challenge ciphertext C_{ID^*,t^*} in Game 2.2:

$$C^1_{ID^*,t^*} = g^{r_1}, C^2_{ID^*,t^*} = g^{r_2},$$
$$C^3_{ID^*,t^*} = (m||r_2) \oplus W^*_{2.1},$$
$$C^4_{ID^*,t^*} = (x_1||x_2||y_1||y_2) \oplus H_6(PK^{r_2}_{test,ID^*,t^*}),$$
$$C^5_{ID^*,t^*} = H_7(H_5(PK^{r_2}_{Rev,ID^*,t^*}), C^1_{ID^*,t^*}, C^2_{ID^*,t^*}, C^3_{ID^*,t^*}, C^4_{ID^*,t^*}).$$

It is because the random oracle is ideal and the hash function is one-way that $\mathcal{A}_2$ can figure out m by the challenge ciphertext with negligible probability ϵ. that is,

$$\Pr[\mathbf{S}_{2.2}] \leq \epsilon. \tag{8}$$

Combining equation $(5) - (8)$, we have that

$$\mathbf{Adv}^{\text{OW-sID-CPA}}_{\text{R-IBEET},\mathcal{A}_2}(\lambda) \leq q_{H_4}\mathbf{Adv}^{\text{CDH}} + \epsilon,$$

which is negligible. The proof of Theorem 2 is complete. □

Theorem 3: The R-IBEET construction is IND-sID-CCA secure against any PPT type-III adversaries provided that CDH assumption holds.

The proof is similar with that of Theorem 1, so we omit it here.

6 Performance analysis

In this section, we analyse the performance of our proposal by comparing it with other related schemes (Yang et al., 2010; Ma et al., 2015; Ma, 2016; Lin et al., 2021b). Among these schemes, Yang et al. (2010) is the first PKEET scheme, Ma et al. (2015) is a PKEET scheme with flexible authorisation, Ma (2016) is the first IBEET scheme and Lin et al. (2021b) is an IBEET scheme with date-stamp authorisation.

In Table 2, the second to fourth rows show the size of public key, ciphertext and trapdoor, respectively. It indicates that in contrast with the other six schemes, the size of our public key and trapdoor is relatively small. The fifth to eighth rows show the computational cost in encryption, decryption, authorisation, test and revocation algorithms, respectively. The result shows that the computational cost of our test algorithm is significantly lower than that of the other four schemes. Despite that, our encryption and decryption algorithms have slightly larger computational cost than those in Yang et al. (2010). The ninth to tenth rows list the security and the assumptions. The last row lists whether or not the schemes have revocation mechanism. As we can see in Table 2, our scheme is the only one that achieves both user and tester revocation. Furthermore, our scheme enjoys especially high test efficiency.

Figure 2 Running time of Enc (see online version for colours)

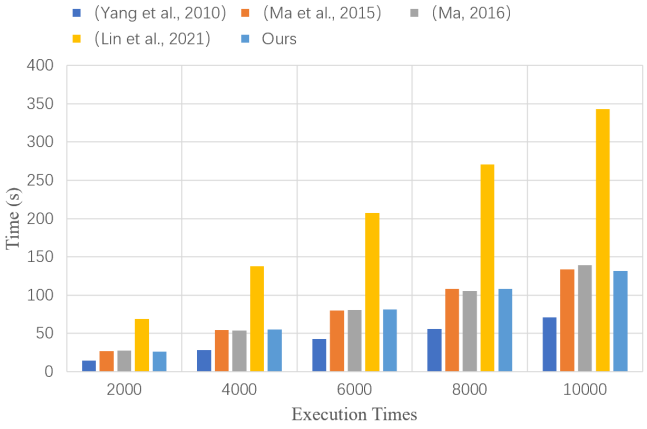


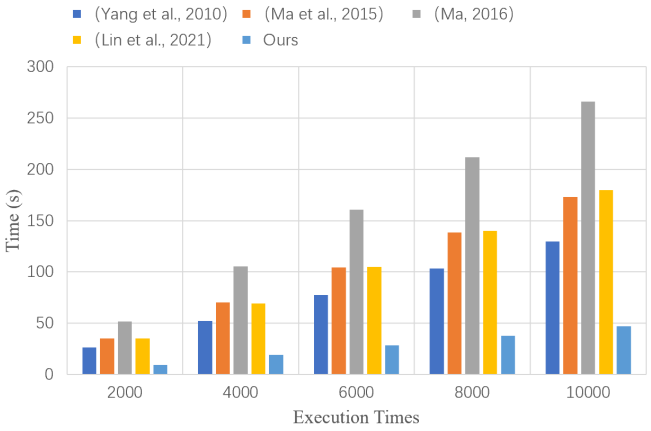
Figure 3 Running time of Dec (see online version for colours)

Table 2 Performance analysis

	Yang et al. (2010)	*Ma et al. (2015)*	*Ma (2016)*	*Lin et al. (2021b)*	*Ours*
$\|PK\|$	$\|\mathbb{G}\|$	$3\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$
$\|C\|$	$3\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	$5\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	$4\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	$3\|\mathbb{G}\| + \lambda$	$3\|\mathbb{G}\| + 5\|\mathbb{Z}_p\| + \lambda$
$\|Td\|$	–	$\|\mathbb{Z}_p\|$	$\|\mathbb{G}\|$	$5\|\mathbb{G}\|$	$\|\mathbb{Z}_p\|$
Enc	3Exp	6Exp	6Exp	9Exp + 2P	6Exp
Dec	3Exp	5Exp	2Exp + 2P	3P	5Exp
Aut	–	0	Exp	7Exp	0
Test	2P	2Exp + 2P	4P	2Exp + 2P	2Exp
Security	OW-CCA	OW-CCA IND-CCA	OW-CCA	OW-CCA IND-CCA	OW-sID-CCA OW-sID-CPA IND-sID-CCA
Assumption	CDH	CDH	BDH	BDHE	CDH
Revocation	×	×	×	×	✓

Notes: Dec – decryption algorithm; Aut – authorisation algorithm; Test – test algorithm; $|\mathbb{G}|, |\mathbb{Z}_p|$ – bit length of an element in $\mathbb{G}$, $\mathbb{Z}_p$, respectively; λ – length of security parameter; $|PK|, |C|, |Td|$ – length of public key, ciphertext and trapdoor, respectively; Exp – modular exponentiation; P – bilinear pairing; BDH – bilinear Diffie-Hellman assumption; CDH – computational Diffie-Hellman assumption; BDHE – bilinear Diffie-Hellman exponent assumption.

Figure 4 Running time of Test (see online version for colours)



To visually show the computational comparison, we implemented the above five schemes using C++ language on a laptop with Intel i5-10600KF 4.10 GHz processor, 16 GB memory and Windows 10 OS. The experiment is based on Multiprecision Integer and Rational Arithmetic C/C++ Library (MiracL) and the results are shown in Figures 2 to 4. The x-axis of Figures 2 to 4 is the execution time, and the y-axis is the total running time. Figure 2 shows that the encryption algorithm of our scheme and other schemes (Ma et al., 2015; Ma, 2016) require comparable running time. Figure 3 shows that the decryption algorithm of our scheme is relatively efficient. Figure 4 shows that our scheme has the most efficient test algorithm among the five schemes, it has reduced the time cost by nearly 82.5%, 73.3% and 63.6% compared with other schemes (Ma, 2016; Lin et al., 2021b; Yang et al., 2010), respectively.

Overall, our scheme is the only one among all the IBEET schemes that can achieve both user and tester revocation. Besides, it achieves efficient test algorithm without sacrificing much efficiency in terms of encryption and decryption.

7 Conclusions

In this paper, we first presented a primitive called revocable identity-based encryption with equality test (R-IBEET), which can achieve both user and tester revocation such that the user is able to get control of the authorisation for the tester, and the PKG is able to revoke the user whenever the user's private key is compromised or identity gets expired. We also proposed a concrete R-IBEET scheme. Based on CDH assumption, we proved its security in random oracle model. Furthermore, our R-IBEET scheme does not need the time-consuming bilinear pairing operations and enjoys especially high test efficiency compared with the related work. However, our scheme only supports user level authorisation, that is, the tester is able to test all of the user's ciphertexts when it is authorised, while in real application scenarios, users may only want the tester to test a certain ciphertext or partial ciphertexts. Obviously, the authorisation method in our scheme is not flexible enough, and we leave devising an R-IBBET scheme with flexible authorisation as our future work.

References

Abdalla, M., Bellare, M., Catalano, D., Kiltz, E. and Shi, H. (2005) 'Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions', *Journal of Cryptology*, Vol. 21, No. 3, pp.350–391.

Alornyo, S., Zhao, Y., Zhu, G. and Xiong, H. (2020) 'Identity based key-insulated encryption with outsourced equality test', *International Journal of Network Security*, Vol. 22, No. 2, pp.257–264.

Boldyreva, A., Goyal, V. and Kumar, V. (2008) 'Identity-based encryption with efficient revocation', *15th ACM Conference on Computer and Communications Security – CCS 2008*, 27–31 October, Alexandria, Virginia, USA.

Boneh, D. and Franklin, M. (2001) 'Identity-based encryption from the Weil pairing', *Annual International Cryptology Conference – CRYPTO 2001*, Proceedings, 19–23 August, Santa Barbara, California, USA.

Boneh, D., Crescenzo, G.D., Ostrovsky, R. and Persiano, G. (2004) 'Public key encryption with keyword search', *Advances in Cryptology – EUROCRYPT 2004, Proceedings*, 2–6 May, Interlaken, Switzerland.

Deebak, B.D., Memon, F.H., Khowaja, S.A., Dev, K., Wang, W. and Qureshi, N.M.F. (2022) 'In the digital age of 5G networks: seamless privacy-preserving authentication for cognitive-inspired internet of medical things', *IEEE Transactions on Industrial Informatics*, Vol. 18, No. 12, pp.8916–8923.

Khowaja, S.A., Khuwaja, P., Dev, K., Lee, I., Khan, W.U., Wang, W., Qureshi, N.M.F. and Magarini, M. (2023) 'A secure data sharing scheme in community segmented vehicular social networks for 6G', *IEEE Transactions on Industrial Informatics*, Vol. 19, No. 1, pp.890–899.

Li, J., Li, J., Chen, X., Jia, C. and Lou, W. (2015) 'Identity-based encryption with outsourced revocation in cloud mputing', *IEEE Transaction on Computers*, Vol. 64, No. 2, pp.425–437.

Libert, B. and Vergnaud, D. (2009) 'Adaptive-ID secure revocable identity-based encryption', *Cryptographer's Track at RSA Conference – CT-RSA 2009, Proceedings*, 20–24 April, San Francisco, CA, USA.

Lin, X., Sun, L. and Qu, H. (2018) 'Generic construction of public key encryption, identity-based encryption and signcryption with equality test', *Information Sciences*, Vol. 453, pp.111–126.

Lin, X., Sun, L., Qu, H. and Zhang, X. (2021a) 'Public key encryption supporting equality test and flexible authorization without bilinear pairing', *Computer Communications*, Vol. 170, pp.190–199.

Lin, X., Wang, Q., Sun, L. and Le, H.Q. (2021b) 'Identity-based encryption with equality test and datestamp-based authorization mechanism', *Theoretical Computer Science*, Vol. 861, pp.117–132.

Ma, S. (2016) 'Identity-based encryption with outsourced equality test in cloud computing', *Information Sciences*, Vol. 328, pp.389–402.

Ma, S., Huang, Q., Zhang, M. and Yang, B. (2015) 'Efficient public key encryption with equality test supporting flexible authorization', *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 3, pp.458–470.

Ming, Y. and Wang, E. (2019) 'Identity-based encryption with filtered equality test for smart city applications', *Sensors*, Vol. 19, No. 14, p.3046.

Qin, B., Deng, R.H., Li, Y. and Liu, S. (2015) 'Server-aided revocable identity-based encryption', *European Symposium on Research in Computer Security – ESORICS 2015, Proceedings*, 21–25 September, Vienna, Austria.

Ramadan, M., Liao, Y., Li, F., Zhou, S. and Abdalla, H. (2020) 'IBEET-RSA: identity-based encryption with equality test over RSA for wireless body area networks', *Mobile Networks and Applications*, Vol. 25, No. 1, pp.223–233.

Shamir, A. (1979) 'How to share a secret', *Communications of the ACM*, Vol. 22, No. 11, pp.612–613.

Sun, Y., Mu, Y., Susilo, W., Zhang, F. and Fu, A. (2020) 'Revocable identity-based encryption with server-aided ciphertext evolution', *Theoretical Computer Science*, Vol. 815, pp.11–24.

Sun, Y., Susilo, W., Zhang, F. and Fu, A. (2018) 'CCA-secure revocable identity-based encryption with ciphertext evolution in the cloud', *IEEE Access*, Vol. 6, pp.56977–56983.

Susilo, W., Duong, D.H. and Le, H.Q. (2020) 'Efficient post-quantum identity-based encryption with equality test', *IEEE 26th International Conference on Parallel and Distributed Systems – ICPADS 2020*, 2–4 December, Hong Kong, SAR, China.

Tang Q. (2012) 'Public key encryption supporting plaintext equality test and user-specified authorization', *Security and Communication Networks*, Vol. 5, No. 12, pp.1351–1362.

Tang, Q. (2011) 'Towards public key encryption scheme supporting equality test with fine-grained authorization', *Australasian Conference on Information Security and Privacy – ACISP 2011, Proceedings*, 11–13 July, Melbourne, Australia.

Wang, C., Zhou, T., Shen, J., Wang, W. and Zhou, X. (2023) 'Searchable and secure edge pre-cache scheme for intelligent 6G wireless systems', *Future Generation Computer Systems*, Vol. 140, pp.129–137.

Wei, J., Liu, W. and Hu, X. (2016) 'Secure data sharing in cloud computing using revocable-storage identity-based encryption', *IEEE Transactions on Cloud Computing*, Vol. 6, No. 4, pp.1136–1148.

Wu, L., Zhang, Y., Choo, K.R. and He, D. (2017) 'Efficient and secure identity-based encryption scheme with equality test in cloud computing', *Future Generation Computer Systems*, Vol. 73, pp.22–312.

Wu, L., Zhang, Y., Choo, K.R. and He, D. (2018) 'Efficient identity-based encryption scheme with equality test in smart city', *IEEE Transactions on Sustainable Computing*, Vol. 3, No. 1, pp.44–55.

Wu, L., Zhang, Y., Choo, K.R. and He, D. (2019) 'Pairing-free identity-based encryption with authorized equality test in online social networks', *International Journal of Foundations of Computer Science*, Vol. 30, No. 4, pp.647–664.

Yang, G., Tan, C., Huang, Q. and Wong, D.S. (2010) 'Probabilistic public key encryption with equality test', *Cryptographers' Track at the RSA conference – CT-RSA 2010, Proceedings*, 1–5 March, San Francisco, CA, USA.

Yu, Y., Ni, J., Yang, H., Mu, Y. and Susilo, W. (2014) 'Efficient public key encryption with revocable keyword search', *Security and Communication Networks*, Vol. 7, No. 2, pp.466–472.

Zhang, J. and Mao, J. (2016) 'Efficient public key encryption with revocable keyword search in cloud computing', *Cluster Computing*, Vol. 19, No. 3, pp.1211–1217.