Cryptanalysis of an Identity-Based Encryption Scheme With Equality Test and Improvement

YONGJIAN LIAO ©, (Member, IEEE), YU FAN, YIKUAN LIANG, YULU LIU, AND RAMADAN MOHAMMED
School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
Corresponding author: Yongjian Liao (liaoyj@uestc.edu.cn)

This work was supported in part by the Sichuan Science and Technology Program under Grant 2018GZ0180, Grant 2017GZDZX0002, Grant 2018GZ0085, and Grant 2017GZDZX0001.

ABSTRACT Privacy-presenting for cloud computing has been concerned for several years. Public key encryption with equality test as a variant of keyword searchable encryption is one of the important concepts in this area. In order to simplify the management of certificate and optimize the scheme, an identity-based encryption with equality test was proposed by Wu et al. In this paper, we analyze that some binary operations of the encryption in the scheme are not clear so that the scheme is not more efficient than the existing schemes or the test of the scheme is not available when we analyze the most possible definitions of the binary operations in the scheme. Finally, we improve the scheme and furthermore, our improved scheme is fine-grained.

INDEX TERMS Identity-based encryption, binary operation, equality test, cloud computing, performance.

I. INTRODUCTION

In cloud computing era, data which are in encrypted form are used to storing in the cloud servers. In order to efficiently manage the data and utilize them in the future, some basic operations on the data for cloud servers are necessary, such as searching the keyword (encrypted form).

Boneh et al. [1] introduced an amazing notion — public key encryption with keyword search (PKEKS), in which an encrypted keyword can be searched by cloud server but cannot be unknown and decrypted. Later, Yang et al. [2] proposed a variant of it — public key encryption with equality test (PKEET), which utilizes the advantage of the public key encryption (PKE) and searchable encryption (SE). The PKEET scheme not only has the functionality of decryption, but also can test whether ciphertexts are encryptions of an unknown keyword even if it is encrypted by different public keys.

Tang proposed an extension of PKEET with fine-grained authorization scheme (PKEET-FG) [3] and all-or-nothing PKEET [4] to improve a PKEET-FG [5]. In order to simplify the certificate management of PKEET, Ma [7] presented identity-based encryption with equality test (IBEET), and showed the scheme is one-way secure under chosen ciphertext attack (OW-CCA). However, the scheme was proved insecure by Liao et al. [8]. In order to improve the efficiency of the IBEET scheme, Wu et al. [9] proposed a new one by reducing the computational cost. Recently, A semi-generic construction of IBEET scheme and PKEET scheme [10] was proposed by Lee et al., but it needs to use the encryption algorithm twice and a one-time signature, which aren’t efficient. Zhang and Xu [11] proposed a scheme from lattices, which is viewed as secure scheme under quantum computing attacks. Chen and Liao [12] considered another properties of ABE scheme and Mohammed et al. [13] proposed IBEET scheme from integer factorization assumption for wireless body area networks.

In this article, we analyze the construction of an IBEET scheme as follows. We first analyze some binary operations used in the IBEET scheme proposed by Wu et al. according to their construction and performance analysis, and find that the definition of a binary operation of $rM$ is not clear, where $r \in \mathbb{Z}_p^*$ and $M \in \{0, 1\}^*$. Because the binary operation of $rM$ is viewed as a usual multiplication in Encryption algorithm and a scalar multiplication in performance analysis, respectively. Then we prove the scheme is inefficient or insecure if we view the binary operation of $rM$ as the usual multiplication, and the Test algorithm cannot perform if we view the binary operation
of $rM$ as the scalar multiplication. Finally, we also improve the IBEET scheme and prove the security and performance. We show that our scheme is more efficient than the existing schemes and is fine-grained.

The article is organized as follows. In section II we introduce some basic notions to be used in the paper. We then recall the model and security model of IBEET in section III. In section IV we recall the IBEET scheme proposed by Wu et al. and analyze it according to different definition of a binary operation, we improve the IBEET scheme and analyze it in section V. Finally, we conclude the paper in section VI.

II. PRELIMINARY

Here, we first recall some basic mathematical knowledge which will be used.

A. BILINEAR PAIRING

Let $G$ be an additive group and $G_T$ be a multiplicative group, which have the same prime order $q$, $Z_q^*$ be the multiplicative group of the finite field $F_q$. A bilinear map $e : G \times G \rightarrow G_T$ [14], which satisfies the following three properties:

- Bilinearity: For any $a, b, c \in G$, $e(a, b + c) = e(a, b)e(a, c)$, and $e(a + b, c) = e(a, c)e(b, c)$.

- Non-degeneracy: For any non-identity elements $g_1, g_2 \in G$, $eg_1, g_2 \neq 1_{G_T}$, where $1_{G_T}$ is the identity element of $G_T$.

- Computability: For any elements $g_1, g_2 \in G$, there is a polynomial-time algorithm to compute $e(g_1, g_2)$.

Definition 1: Computational Diffie-Hellman problem (CDH problem). Let $G$ be an additive group above. Given $(P, aP, bP) \in G^3$ for some $a, b \in Z_q^*$, to compute $abP$.

B. MODEL OF IBEET

Three parties are in an IBEET as follows, the key generator center (KGC), user and the cloud server, which are in FIGURE 1. The KGC produces the user’s private key. The user produces the trapdoor of the private keys and ciphertexts. The cloud server can store the data of users which are in encrypted form and run the test algorithm when it receives the ciphertexts’ trapdoors. The user can get its private key over a secure channel. The cloud server can get data (ciphertexts) and trapdoors over open channels.

An IBEET scheme [9] consists of six algorithms as follows.

- **Setup($k$)**: It produces public parameters $PKT$ and a master key $msk$ for the input a security parameter $k$.

- **Extract($msk$, $ID$)**: It produces a private key $sk_{ID}$ of an identity $ID$ for the input $msk$ and the identity $ID \in \{0, 1\}^*$.

- **Enc($ID$, $M$)**: It produces a ciphertext $C \in \mathcal{C}$ for the input $ID \in \{0, 1\}^*$ and a message $M \in \mathcal{M}$. Where $\mathcal{M}$ and $\mathcal{C}$ are the plaintext space and ciphertext space, respectively.

- **Dec($sk_{ID}$, $C$)**: It produces a plaintext $M \in \mathcal{M}$ for a ciphertext $C \in \mathcal{C}$ and a private key $sk_{ID}$.

- **Trapdoor($sk_{ID}$, $C$)**: It produces a trapdoor $td_{A}$ for a user $A$’s private key $sk_{ID}$ and a ciphertext $C$ which is encrypted by using $ID_{A}$. It produces the same trapdoor $td_{A}$ for all ciphertext $C$ if $ID_{A}$ is an empty string.

- **Test($C_{A}$, $td_{A}$, $C_{B}$, $td_{B}$)**: It outputs “1” if the plaintext of $C_{A}$ and $C_{B}$ are the same; Otherwise it outputs “0”.

If the trapdoors for the ciphertexts are different, we call the IBEET scheme is fine-grained.

C. SECURITY MODEL OF IBEET

We recall the definition of one-way against chosen ciphertext security (OW-ID-CCA) for IBEET scheme [9].

- **Setup** The challenger $C$ runs the Setup algorithm to produce the public parameters $PKT$ and master key $msk$. Then it sends $PKT$ to the adversary $A$.

- **The Phase 1**

  - Private-key-query: $C$ runs the Extract algorithm to produce the private key $sk_i$ of an identity $ID_i$. Then it sends the private key $sk_i$ as a response of $ID_i$’s query to adversary $A$.

  - Trapdoor-query $TD_{i}$. $C$ runs the above private-key-query to produce the trapdoor $td_{i}$ at any time, and then sends it to $A$.

  - Decryption-query ($ID_{i}, C_{i}$). $C$ runs the decryption oracle to decrypt the ciphertext $C_{i}$, and then sends an output of the decryption oracle, $M_{i}$, to adversary $A$.

- **Challenge**: $A$ submits a challenge identity $ID^*$ which didn’t appear in the private-key-query in the phase 1. Then $C$ picks a plaintext $M^* \in \mathcal{M}$ randomly and sends $C^* = Enc(ID^*, M^*)$ to $A$ as a challenge ciphertext.

- **The Phase 2.** All queries are the same as them in the Phase 1, except

  - $ID_{i} \neq ID^*$ in the Private-key-query.

  - $(ID_{i}, C_{i}) \neq (ID^*, C^*)$ in the Decryption-query.

- **Guess**: $A$ submits a guess $M’ \in \mathcal{M}$.

$A$ is called a OW-ID-CCA adversary in the above game [9]. The OW-ID-CCA adversary’s advantage is the probability which $A$ wins the game, i.e.,

$$Ad^{OW-ID-CCA}_{IBEET,A}(k) = Pr[M = M’].$$
Definition 2: We call an IBETE scheme to be OW-ID-CCA secure if $Adv_{IBETE,A}^{OW-ID-CCA}(k)$ is negligible in $k$ for all OW-ID-CCA adversaries.

III. CRYPTOANALYSIS OF AN IBETE SCHEME
In this section, we recall the IBETE scheme proposed by Wu et al. [9] before we analyze it.

A. REVIEW OF THE IBETE SCHEME
- Setup: On input a security parameter $k \in \mathbb{Z}^{+}$, the algorithm generates the following system public parameters $K$. $G$ and $G_T$ are an additive group and a multiplicative group with the same prime order $p \in \mathbb{Z}^{+}$ respectively. Let $e : G \times G \rightarrow G_T$ be a bilinear map, and $P$ be a generator of group $G$. Set $g = e(P, P)$. $H : G_T \rightarrow G$, $h_1 : \{0, 1\}^* \rightarrow \{0, 1\}^*$ and $h_2 : G_T \rightarrow \{0, 1\}^*$ are three hash functions. The algorithm then randomly picks two numbers $s, s' \in \mathbb{Z}_p^*$, and computes $P_{pub} = sP$, $P'_{pub} = s'P$. Finally, it publishes

$$K = (q, G, G_T, P, e, g, P_{pub}, P'_{pub}, H, h_1, h_2)$$

as public parameters and $(s, s')$ as the master key.
- Extract: On input an identity $ID \in \{0, 1\}^*$, the algorithm computes $h_{ID} = h_1(ID)$ and private key $sk_{ID} = (sk_{1,ID}, sk_{2,ID}) = (\frac{1}{h_{ID}+s}P, \frac{1}{h_{ID}+s'}P)$.
- Enc: On input an identity $ID \in \{0, 1\}^*$ and a plaintext $M \in \{0, 1\}^*$, the algorithm randomly picks two numbers $r_1, r_2 \in \mathbb{Z}_p^*$, and computes the ciphertext $C = (C_1, C_2, C_3, C_4)$, where

$$C_1 = r_1(h_{ID}P + P'_{pub}),$$
$$C_2 = (r_1M) \oplus H(g^{r_1}),$$
$$C_3 = r_2(h_{ID}P + P_{pub}),$$
$$C_4 = (M||r_1) \oplus h_2(g^{r_2}).$$

- Dec: On input ciphertext $C = (C_1, C_2, C_3, C_4)$ and the private key $sk_{ID} = (sk_{1,ID}, sk_{2,ID})$ of the identity $ID$, the algorithm computes

$$C_4 \oplus h_2(e(\frac{1}{h_{ID}+s}P, C_3)) = M||r_1,$$

then it verifies

$$C_1 = r_1(h_{ID}P + P'_{pub})$$

and

$$C_2 \oplus r_1M = H(e(\frac{1}{h_{ID}+s'}P, C_1)).$$

If both equations hold, then the algorithm outputs $M$.
- Trapdoor: On input an identity $ID \in \{0, 1\}^*$, the algorithm computes the trapdoor $td_{ID} = sk_{2,ID} = \frac{1}{h_{ID}+s}P$.
- Test: On input $(C_1, td_{ID_A}, C_2, C_3, C_4)$, where

$C_A = (C_{1,A}, C_{2,A}, C_{3,A}, C_{4,A}) = Encrypt(ID_A, M_A)$,
$C_B = (C_{1,B}, C_{2,B}, C_{3,B}, C_{4,B}) = Encrypt(ID_B, M_B)$, $td_{ID_A} = \frac{1}{h_{ID_A}+s'}P$ and $td_{ID_B} = \frac{1}{h_{ID_B}+s'}P$.

The algorithm computes

$$E_A = e(td_{ID_A}, C_{1,A}) = e(\frac{1}{h_{ID_A}+s'}P, C_{1,A}),$$
$$X_A = C_{2,A} \oplus H(E_A)$$
$$E_B = e(td_{ID_B}, C_{1,B}) = e(\frac{1}{h_{ID_B}+s'}P, C_{1,B}),$$
$$X_B = C_{2,B} \oplus H(E_B)$$

and verifies

$$(E_A)^{X_A} = (E_B)^{X_B}.$$ 

If the above equation holds, then $M_A = M_B$.

B. ANALYSIS OF THE SCHEME
The definition of a binary operation of $rM$ wasn’t clear in [9]. On the one hand, the binary operation of $rM$ can be viewed as the normal multiplication of two fixed size numbers (strings) from the construction of the Enc algorithm, where $r \in \mathbb{Z}_p^*$ and $M \in \{0, 1\}^*$. On the other hand, the binary operation of $rM$ had been viewed as a scalar multiplication in the performance in [9] (They countered that there are 5 scalar multiplications in the Enc algorithm). At the same time, $rM$ had been viewed as a scalar multiplication in almost all other PKEET and IBETE schemes [2], [6], [7]. Next, we analyze the IBETE scheme from the two possible definitions of the binary operation, i.e., a binary operation of $rM$ is an integer multiplication and a scalar multiplication, respectively.

1) BINARY OPERATION OF $rM$ IS AN INTEGER MULTIPLICATION
If the definition of the binary operation of $rM$ is viewed as multiplication of two integers (strings), then the scheme will produce inefficiency, and furthermore it will cause some security problem. We analyze it as follows.

One of contribution of the scheme [9] is that the scheme was efficient and available to mobile computing (in subsection 1.1 of [9]). They claimed their scheme was more efficient than the IBETE scheme proposed by Ma [7]. However, we analyze their scheme and find this advantage is not correct if we take into account the usual multiplication of $rM$. From the game of the definition 2, any adversary can make trapdoor query to obtain trapdoor $td$ of the challenge ciphertext. Thus, the adversary has

$$r_1^*M^* = C_2^* \oplus H(e(td, C_1^*)).$$

On the one hand, we analyze the efficiency of their IBETE scheme. For the security perspective, the length of $r_1^*M^*$ must be more than 1024 bits (For RSA scheme, the security parameter is at least 1024 bits), that implies the group $G$ over a finite field $\mathbb{Z}_p^*$ is more than 512 bits. But a secure IBETE scheme [8] which are improved from the IBETE scheme proposed by Ma [7] is secure over a finite field $\mathbb{Z}_p^*$ about 160 bits. We know that computational cost of schemes over elliptic curve of 512 bits are much less than it of schemes over elliptic curve of 160 bits. Thus, their scheme [9] is much less
efficient than the Ma’s IBEET scheme if viewing the binary operation of $rM$ as the usual integer multiplication.

On the other hand, $r_1^* \in \mathbb{Z}_p^*$ and $M^* \in \{0, 1\}$ are random but not prime, and there exists risk of factorizing $r_1^*M^*$ even the length of $r_1^*M^*$ is more than 1024 bits. Once the adversary factors it, it can get $r_1^*$ from testing the equation

$$C_1^* = r_1^*(hDP + P')_p,$$

where $r_1^*$ is an element of set of all possible divisors of $r_1^*M^*$. That implies $r_1^* = r_1^*$. Then the adversary can compute

$$M^* = r_1^{-1}r_1^*M^*.$$  

Thus, the scheme maybe not satisfy the definition 2, one-way security if viewing the binary operation of $rM$ as the usual integer multiplication. Its security relies on the choice of $r$ and $M$. Additionally, it will produce an additional cost (encoding $M$) if the scheme requires that the ‘keyword’ $M$ is prime.

2) **Binary Operation of $rM$ is a Scalar Multiplication**

In this subsection, the binary operation of $rM$ is taken into account as the scalar multiplication, which was used in the Test algorithm of the IBEET scheme and viewed as a scalar multiplication in the performance [9].

Suppose that $\mathbb{G}$ is an additive group of some elliptic curve and $M \in \mathbb{G}$ is a point of the elliptic curve, which is defined in Section II. Let $\mathbb{Z}_+^*$ be the set of positive integers. Thus, for a random number $r \in \mathbb{Z}_+^*$, we have

$$r \cdot M \overset{\text{Def}}{=} M + \cdots + M.$$  

Here, the multiplication is the scalar multiplication over the group $\mathbb{G}$.

However, the point $M'$ in $\mathbb{G}$ and the number $r' \in \mathbb{Z}_p^*$ are viewed as bit strings in the IBEET scheme [7], [9] and the PKEET scheme [2], [4]. Because the coordinates of $M'$ (set $M' = (m_1, m_2)$) are elements of a finite field, which can be viewed as a concatenation of two bit strings, i.e. $m_1||m_2$. Thus, the definition of the multiplication over two bit strings is described as follows.

$$r' \cdot M' \overset{\text{Def}}{=} r' \times M'.$$

where the symbol “$\times$” is the multiplication symbol of the usual multiplication on the integer set $\mathbb{Z}$.

Next, we take into account the following equality which was used in [9].

$$r' \cdot (r \cdot M) \overset{\text{Def}}{=} r \cdot (r' \cdot M).$$

Obviously, although $M$ is an element of $\mathbb{G}$ and $\mathbb{G}$ is an abelian group, the binary operations $\circ$ and $\cdot$ are not the same operations, the ‘associative’ law and the ‘commutative’ law do not hold. That is to say,

$$r' \cdot (r \cdot M) \neq r \cdot (r' \cdot M).$$

In order to explain the above inequality clearly, we give the following example to show it is correct.

Let $\mathcal{EC} : y^2 = x^3 + 7x$ be an elliptic curve over a finite field $\mathbb{Z}_{13}$. It is easy to verify $(3, 3)$ is a point of order 3 over the elliptic curve $\mathcal{EC}$. Suppose that $r = 1$, $r' = 2$ and $M = (3, 3)$. We have

$$r' \cdot (r \cdot M) = 2 \cdot (1 \cdot (3, 3)) = 2 \cdot (3, 3) = 30,$$

$$r \cdot (r' \cdot M) = 1 \cdot (2 \cdot (3, 3)) = 1 \cdot (3, 10) = 58.$$  

where we encode 3 as a bit string 11 and encode 10 as a bit string 1010. So $(3,3)$ is 1111, $(3,10)$ is 111010.

Thus, we have $r' \cdot (r \cdot M) \neq r \cdot (r' \cdot M)$.

Maybe there are another encoding algorithm to encode a point in $\mathbb{G}$. However, if we define the bilinear pairing $\hat{e}$ with two multiplicative groups $\mathbb{G}$ and $\mathbb{G}_T$ with the same order $q$,

$$\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T.$$  

$r' \cdot (r \cdot M)$ should be written as $r' \cdot M'$, it is obvious that $r' \cdot M' \neq r \cdot M'$. In fact, the IBEET scheme proposed by Wu et al. had misused these two binary operations in their Test algorithm.

**Analysis of Their IBEET Scheme**

Next, we will show that the construction of the IBEET scheme is not rational, which causes the Test algorithm cannot be performed.

Since the Test algorithm needs to verify the equality

$$(E_A)^{X_B} = (E_B)^{X_A},$$

where $E_A = g^{r_1A} \in \mathbb{G}$, $E_B = g^{r_1B} \in \mathbb{G}$, $X_A = r_1AM$ and $X_B = r_1BM$. That means that the Test algorithm needs to verify the equality

$$g^{r_1A(r_1BMB)} = g^{r_1B(r_1AMA)}$$

to determine if that $M_A = M_B$ holds or not. That is to say, $r_1A(r_1BMB) = r_1B(r_1AMA) \mod p$ holds if and only if $M_A = M_B$.

However, as we discuss in previous subsection, this equality doesn’t hold. Because for $r_1A, r_1B \in \mathbb{Z}_p^*$ and $M_A, M_B \in \mathbb{G}$, in the above equality the binary operation of $r_1B$ and $r_1AM$ is the scalar multiplicative operation over $\mathbb{G}$, but the binary operation of $r_1A$ and $(r_1BM)$, $r_1B$ and $(r_1AM)$ is the multiplicative operation over $\mathbb{Z}_p^*$. On the one hand, $r_1M$ is not $r_1 \cdot M$ defining in the subsection A of this section. If $r_1M$ is $r_1 \cdot M$, the Enc algorithm cannot control the length of $r_1M$, and which causes the operation of $C_2 = (r_1M) \oplus H(g^{r_1})$ cannot run. On the other hand, the equality $(E_A)^{X_B} = (E_B)^{X_A}$ in the Test algorithm means that $X_A, X_B$ are viewed as numbers, but not elements of $\mathbb{G}$.

Thus these two “multiplications” are not the same binary operations, and the associative law and the commutative law do not hold. Thus, even though the equality $M_A = M_B$ holds, the inequality

$$g^{r_1A(r_1BMB)} \neq g^{r_1B(r_1AMA)}$$

holds.
IV. OUR IMPROVED SCHEME

In this section, we will improve the IBEET scheme proposed by Wu et al. as follows. Where we view the binary operation of $rM$ as the scalar multiplication for $r \in \mathbb{Z}_p$ and $M \in G$.

A. OUR CONSTRUCTION

We do not change Setup, Extract and Decrypt algorithms of their IBEET scheme, almost not change the $\text{Enc}$ algorithm except that $M \in G$ and hash functions $H : \mathbb{G}_T \rightarrow \mathbb{G}$, $h_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ and $h_2 : \mathbb{G}_T \rightarrow \{0,1\}^{[\mathbb{G}_T+|q|}}$ are required. We describe the other algorithms of the IBEET scheme as follows.

- **Enc**: On input an identity $ID \in \{0,1\}^*$ and a plaintext $M \in G$, the algorithm randomly picks two numbers $r_1, r_2 \in \mathbb{Z}_p$, and computes the ciphertext $C = (C_1, C_2, C_3, C_4)$, where
  
  
  \[
  C_1 = r_1(h_{ID}P + P_{pub}), \\
  C_2 = (r_1 M) \oplus H(g^{r_1}), \\
  C_3 = r_2(h_{ID}P + P_{pub}), \\
  C_4 = (M||r_1) \oplus h_2(g^{r_2}).
  \]

- **Trapdoor**: On input an identity $ID \in \{0,1\}^*$ and its private key $(sk_{1,ID}, sk_{2,ID})$, the algorithm computes
  
  \[
  C_4 \oplus h_2(e(sk_{1,ID}, C_3)) = M||r_1,
  \]

  and then outputs the trapdoor $td_{ID} = (td_{1,ID}, td_{2,ID}) = \left(\frac{1}{r_{ID}+\epsilon} P, r_1 P\right)$.

- **Test**: On input $(C_A, td_{ID_A}, C_B, td_{ID_B})$, where
  
  \[
  C_A = (C_{1,A}, C_{2,A}, C_{3,A}, C_{4,A}) = \text{Encrypt}(ID_A, M_A), \\
  C_B = (C_{1,B}, C_{2,B}, C_{3,B}, C_{4,B}) = \text{Encrypt}(ID_B, M_B),
  \]

  the algorithm computes

  \[
  E_A = e(td_{1,ID_A}, C_{1,A}) = e(\frac{1}{h_{ID_A} + \epsilon} P, C_{1,A}), \\
  X_A = C_{2,A} \oplus H(E_A), \\
  E_B = e(td_{1,ID_B}, C_{1,B}) = e(\frac{1}{h_{ID_B} + \epsilon} P, C_{1,B}), \\
  X_B = C_{2,B} \oplus H(E_B)
  \]

  and verifies

  \[
  e(td_{2,ID_A}, X_A) = e(td_{2,ID_B}, X_A).
  \]

If the above equation holds, then $M_A = M_B$.

Our Test algorithm is distinct to it in [7]. In order to run the Test algorithm, our Trapdoor algorithm should generate a corresponding trapdoor for every ciphertext.

B. SECURITY ANALYSIS OF OUR IMPROVED IBEET SCHEME

We first show the **Correctness** of the improve scheme. On input the trapdoors $td_{ID_A}, td_{ID_B}$ and the ciphertexts $C_A, C_B$, the algorithm Test can compute:

\[
X_A = C_{2,A} \oplus H(E_A) = C_{2,A} \oplus H(e(td_{1,ID_A}, C_{1,A})) = C_{2,A} \oplus H(g^{r_1}) = r_1 M_A,
\]

and

\[
X_B = C_{2,B} \oplus H(E_B) = C_{2,B} \oplus H(e(td_{1,ID_B}, C_{1,B})) = C_{2,B} \oplus H(g^{r_1}) = r_1 M_B.
\]

We have

\[
e(td_{2,ID_A}, X_A) = e(r_1 A P, r_1 B M_B) = e(P, M_B)^{r_1 A r_1 B},
\]

\[
e(td_{2,ID_B}, X_A) = e(r_1 A P, r_1 A M_A) = e(P, M_A)^{r_1 A r_1 B}.
\]

Thus, the equality

\[
e(td_{2,ID_A}, X_B) = e(td_{2,ID_B}, X_A)
\]

holds if and only if $M_A = M_B$.

Our improved scheme is based on the IBEET scheme proposed by Wu et al., and we do not change Setup, Extract, Encrypt and Decrypt algorithms of their scheme. But in order to perform the equality test, we improve the Trapdoor algorithm and Test algorithm. The trapdoor $td_2$ is the one and only if the Encrypt algorithm had generated the ciphertext $(C_1, C_2, C_3, C_4)$ of some plaintext $M$. At the same time, $td_2$ does not reveal any information on the plaintext $M$ and the random numbers $r_1$ and $r_2$. That means these modifies do not change the security of the IBEET scheme. Thus, we get the following Theorem 1, which can be proved by using the same method in [9].

**Theorem 1**: Suppose a OW-ID-CCA adversary $A$ has advantage $\epsilon(k)$ against our improved IBEET scheme, then there also exists an algorithm that can solve the CDH problem in group $G$ with advantage of at least

\[
\epsilon(k) = \frac{(q_H + q_{h_2} + q_{dec})e(q_{sk} + q_{td} + q_{dec} + 1)}{(q_{dec}q_H/2^l(q_H + q_{h_2} + q_{dec}) + (1/2^l + 1/2^l)(q_{h_2} + q_{dec})}
\]

where $q_{H}, q_{h_2}, q_{dec}, q_{sk}$ and $q_{td}$ denote the number of hash function $H$ queries, hash function $h_2$ queries, decryption queries, private key queries and trapdoor queries, respectively.

Since the proof of the above theorem is almost the same as the proof in [9], we omit it here.

C. PERFORMANCE AND COMPARISON

Because we do not modify the Setup, Extract, Encrypt and Decrypt algorithms of the IBEET scheme proposed by Wu et al., it is easy to know that the computation cost of the
In this paper, we firstly proved that some binary operation used in the IBEET scheme proposed by Wu et al. was not reasonable, which caused the scheme less efficient or insecure or non-available according to different definitions. Then we improved the scheme and constructed a secure and efficient IBEET scheme which is fine-grained.

**ACKNOWLEDGEMENT**

The authors would like to thank the anonymous reviewers for their very valuable comments.

**REFERENCES**


**TABLE 1.** Comparison of IBEET Schemes.

<table>
<thead>
<tr>
<th>Schemes</th>
<th>[7]</th>
<th>[9]</th>
<th>[10]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enc Cost</td>
<td>$4E_G+2E_{ct}$</td>
<td>$4E_G+2E_{ct}$</td>
<td>$3E_G+2E_{ct}$</td>
<td>$4E_G+2E_{ct}$</td>
</tr>
<tr>
<td>Dec Cost</td>
<td>$2BP+2E_G$</td>
<td>$2BP+2E_G$</td>
<td>$2BP+3E_G$</td>
<td>$2BP+2E_G$</td>
</tr>
<tr>
<td>Test Cost</td>
<td>$4BP$</td>
<td>$2BP+2E_G$</td>
<td>$2BP$</td>
<td>$2BP+3E_G$</td>
</tr>
<tr>
<td>Length of MSK</td>
<td>$1E_G$</td>
<td>$1E_G$</td>
<td>$1E_G$</td>
<td>$1E_G$</td>
</tr>
<tr>
<td>Length of CT</td>
<td>$4[G]+[H]$</td>
<td>$3[G]+[H]$</td>
<td>$4[G]+[H]$</td>
<td>$3[G]+[H]$</td>
</tr>
<tr>
<td>Length of TD</td>
<td>$[G]$</td>
<td>$[G]$</td>
<td>$[G]$</td>
<td>$[G]$</td>
</tr>
<tr>
<td>Function of Test</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Security</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Fine-Grained [5]</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
</tbody>
</table>

Where Enc Cost, Dec Cost and Test Cost represent the computation cost of encryption, decryption and test, respectively. Length of MSK, CT and TD represent the length of the master key, ciphertext and trapdoor, respectively. $E_G$, $E_{ct}$, and $BP$ represent the average computation cost of an exponentiation of $G$ and $CT$ and bilinear pairing, respectively. $[G]$, $[G_T]$ and $[Z_P]$ represent the average length of an element of $G$, $G_T$ and $Z_P$, respectively. $[H]$ and $h$ represent the length of the output of the hash functions $H$ and $h$, respectively.