An IBE Scheme with Verifiable Outsourced Key Generation Based on a Single Server

Ting Xue, Yanli Ren and Guorui Feng

School of Communication and Information Engineering, Shanghai University, Shanghai 200444, People's Republic of China

ABSTRACT

It is well known that private key generation is a computing bottleneck in an identity-based encryption scheme when a large number of users exist. Outsourcing computation can greatly reduce the users' computational cost, but the existing schemes are all based on two servers, which are not feasible in the real cloud environment. In this paper, we first propose a scheme where the PKG outsources the task of private key generation to a single server, and the results can be verified effectively. Moreover, PKG only needs to execute 1 modular exponentiation, so it is more efficient when compared with previous schemes. Meanwhile, we prove the indistinguishability of the ciphertext and verifiability of the outsourcing results in the security model. Finally, the proposed algorithm is realized in a simulation environment. Theoretical analysis and experimental results show that total computing time of PKG and the server in the outsourced algorithm is far less than that of direct computation.

KEYWORDS

Identity-based encryption; Private key generation; Verifiable outsourcing computation

1. INTRODUCTION

With the rapid development of cloud computing, the safety and efficiency of outsourcing computation have attracted widespread attention. By outsourcing computation, outsourcers with limited ability can outsource complex computing tasks to powerful servers, which will greatly reduce users' computational costs. However, outsourcing computing tasks to servers also presents some security risks because the servers are not completely trusted. Verifiable outsourcing schemes can keep the information of users confidential and verify computational results returned from the server. At present, verifiable outsourcing schemes have involved in many aspects in cryptography, such as modular exponentiation [1,2], polynomial operations [3], attribute-based encryption operations [4,5], image feature extraction [6], and so on.

Modular exponentiation is one of the most time-consuming and commonly used operations in cryptography, and many outsourcing schemes for modular exponentiations have been proposed. Hohenberger et al. [7] first proposed an outsourcing scheme for single modular exponentiation with two non-colluding servers, where the outsourcer could check the fault that the servers might make with a probability of 1/2, and they also defined the security model of outsourcing calculation. Then Wang et al. [8] presented an outsourcing solution for batch modular exponentiations based on a single untrusted server with increased security, but the checkability was only 1/(s + 1)(where s is the number of modular exponentiations), which means it was not efficient and the checkability was only 1/2 for outsourcing single modular exponentiation. Different from the previous schemes, Liu et al. [9] did the work for outsourcing composite modular exponentiations. Ye et al. [10] proposed a new secure outsourcing algorithm based on a single server, and the verifiable probability is close to 1 while the outsourcer appended much computation cost. Chen et al. [11] firstly presented an efficient outsourcing algorithm for bilinear pairing based on two untrusted servers and the outsourcer need not any expensive operations in their algorithm. Then another outsourcing algorithm was proposed with improved checkability based on two servers for bilinear pairing [12]. Recently, Ren et al. [13] presented a new outsourcing algorithm of bilinear pairing with a verifiability of close to 1 if one of the servers misbehaved, which improved the checkability without an interactive operation between the server and the outsourcer.

In identity-based cryptography, the user’s identity is directly used as a public key, and the private key generator needs to generate a private key or perform key update for each user. The identity-based cryptography was firstly proposed in 1984 by Shamir [14], but the first IBE scheme was constructed by Boneh and Franklin in 2001 [15]. In order to reduce the risk of key leakage,
Ren et al. [16] proposed an identity-based parallel key encryption scheme, then Yu et al. [17] presented a securer solution because this algorithm supports frequent key updates without increasing the risk of key leakage. With the increase of users, the calculation burden of PKG will be heavy and it may cause the bottleneck of the system. In order to solve this problem effectively, Li et al. [18] proposed an outsourced deletion algorithm based on identity encryption scheme, which transfers the calculation burden of deleting users from PKG to cloud server. Recently, Ren et al. [19] firstly proposed a verifiable algorithm for outsourcing private key generation in the IBE scheme. Based on the LW — IBE scheme [20], PKG can outsource the tasks of generating private keys and verify the correctness of returned results. This scheme can verify the results with a probability of 1 if one of the servers returned fault results, but it is based on two untrusted servers. So far, there is no relevant algorithm proposed based on a single server.

Our contributions: In this paper, we introduce the concept of verifiable computation into the identity-based encryption scheme, and then propose an IBE scheme that includes an outsourcing private key generation algorithm. In the proposed scheme, PKG outsources the tasks of generating the private key and verifies the returned results effectively. The outsourcing algorithm is based on a single server and it can greatly reduce the computational cost of PKG, meanwhile, the server cannot obtain PKG or the private key of any user during the outsourcing process. The PKG can detect the failure with a probability of about 1 if the server misbehaves, the experiment shows the time cost for PKG is much smaller than that of generating the private keys directly and the server in the outsourcing algorithm.

2. DEFINITION AND SECURITY MODEL

In this section, we introduce some definitions including bilinear groups with the order of composite number, the identity-based encryption scheme and the security model of IBE.

2.1 Bilinear Groups with Order of Composite Number

People first presented bilinear groups with the order of composite number in [20], the algorithm takes security parameter $\lambda$ as input and then outputs $(N = p_1 p_2 p_3, G, G_T, e)$, where $p_1, p_2, p_3$ are distinct primes, $G$ and $G_T$ are cyclic groups with order $N = p_1 p_2 p_3$, if it has following conditions, then $e: G \times G \to G_T$ is a bilinear map:

1. (Bilinear): $\forall g, h \in G, a, b \in Z_N, e(g^a, h^b) = e(g, h)^{ab}$.
2. (Non-degenerate): $\exists g \in G$, which makes $e(g, g)$ is a generator of $G_T$.
3. (Computable): There is an efficient algorithm to compute $e(g, h)$ for all $g \in G, h \in G$.

Let $G_{p_1}, G_{p_2}, G_{p_3}$ represent the subgroups of $G$ with order $p_1, p_2, p_3$, respectively. $G_{r_1p_1}, G_{r_2p_2}, G_{r_3p_3}$ represent the subgroups of $G_T$ with order $p_1, p_2, p_3$, respectively. So $G = G_{p_1} \times G_{p_2} \times G_{p_3}$, $G_T = G_{r_1p_1} \times G_{r_2p_2} \times G_{r_3p_3}$. Assume $g$ is a generator of $G$, and then $g^{p_2p_3}, g^{p_1p_3}, g^{p_1p_2}$ are generators of $G_{p_1}, G_{p_2}, G_{p_3}$, respectively, $\forall h_i \in G_{p_i}, h_j \in G_{p_j}$ and $i \neq j$, we have $e(h_i, h_j) = 1$. If $h_1 \in G_{p_1}$, $h_2 \in G_{p_2}$, since that $g^{p_1p_2p_3} = g^N = 1$, then, for some $a$ and $b, h_1 = (g^{p_2p_3})^a$, $h_2 = (g^{p_1p_3})^b$, and $e(h_1, h_2) = (g^a)^{p_1} (g^b)^{p_2p_3} = 1$.

2.2 Identity-Based Encryption Scheme

An IBE scheme is usually composed of four aspects, including Setup, KeyGen, Encrypt, and Decrypt algorithms [20], which typically involves two entities, PKG, and users (including sender and receiver). If PKG sends the task of outsourcing private key generation to the server, the KeyGenOut algorithm will be added and the scheme consists of the following algorithms.

Setup ($\lambda$): The setup algorithm takes a security parameter $\lambda$ as input, and outputs the master key $MK$ and the public key $PK$. The key $PK$ is public and the master key is kept secret.

KeyGen ($MK, ID, PK$): The private key generation algorithm takes the public key $PK$, an identity ID and the master key $MK$ as input, and outputs a private key $SK_{ID}$ corresponding to the identity ID.

Encrypt ($M, ID, PK$): The encryption algorithm takes an identity $ID$, a public key $PK$ and a message $M$ as input, and outputs the ciphertext $CT$.

Decrypt ($CT, SK_{ID}, PK$): The decryption algorithm takes the ciphertext $CT$, a private key $SK_{ID}$ and the public key $PK$ as input, and it returns a message $M$ or an error $\bot$.

If PKG outsources the task of generating private keys to the server, then the following algorithm is appended. The computation model of the scheme is shown in Figure 1.

KeyGenOut($ID$): The algorithm takes the identity $ID$ as input, PKG generates outsourcing private key $OK_{ID}$ and
sends it to the server. When the server returns the calculation results \( CR_{ID} \), we verify the results correct or not, if correct, then calculate the user’s private key \( SK_{ID} \), otherwise return an error \( \bot \).

2.3 Security Model of an IBE Scheme

We first introduce the full security model against the chosen-plaintext attack (CPA) in an IBE scheme, and then introduce the security model of an IBE scheme which adds the outsourcing private key generation algorithm. A game between a challenger and an adversary \( A \) can describe them.

In the following games, we define the advantage of \( A \) and the scheme is secure if the advantage is negligible.

**Definition 1:** (IND-ID-CPA) Through the following games, the definition of indistinguishability of an IBE scheme under the chosen-plaintext attack is given [20]. The game contains 2 participants: challenger and adversary \( A \).

**Setup:** The challenger executes the Setup algorithm to get the public key \( PK \) and the master key \( MK \), which then sends the public key \( PK \) to attacker \( A \) while ensuring that \( MK \) is secret.

**Phase 1:** First the challenger initializes an empty outsourcing list \( OL \) and the adversary \( A \) adaptively issues queries.

Private key query: the adversary \( A \) sends an identity \( ID \) to the challenger, then the challenger runs the KeyGen algorithm, and sends the private key of identity to the adversary \( A \).

Outsourcing key query: The adversary \( A \) sends \( ID \) to the challenger and then the challenger search \((ID, OKID)\) in \( OL \). If it exists, then returns \( OKID \) to \( A \), Otherwise, generates \( OKID \) and returns it to \( A \).

**Challenge:** A sends an identity \( ID^* \) and two same length messages \( M_0, M_1 \) to the challenger, in Phase 1, \( ID^* \) has not been executed the private key query or outsourcing key query. The challenger randomly chooses a bit \( \beta \in \{0, 1\} \) and forms the ciphertext \( CT^* = \text{Encrypt}(PK, M_\beta, ID^*) \), and then returns \( CT^* \) to \( A \).

**Phase 2:** The adversary \( A \) continues to issue private key queries and the challenger responds queries adaptively. However, \( ID^* \) will not been executed the private key query.

**Guess:** A outputs guess \( \beta' \in \{0, 1\} \).

**Definition 2:** (IND-OID-CPA) Through the following games, the definition of indistinguishability of an IBE scheme under the chosen-plaintext attack is given after adding private key generation algorithm [19]. The game contains two participants: challenger and adversary \( A \).

**Setup:** The challenger executes the Setup algorithm to get the public key \( PK \) and the master key \( MK \), which then sends the public key \( PK \) to attacker \( A \) while ensuring that \( MK \) is secret.

**Phase 1:** First the challenger initializes an empty outsourcing list \( OL \) and the adversary \( A \) adaptively issues queries.

Private key query: the adversary \( A \) sends an identity \( ID \) to the challenger, then the challenger runs the KeyGen algorithm, and sends the private key of identity to the adversary \( A \).

Outsourcing key query: The adversary \( A \) sends \( ID \) to the challenger and then the challenger search \((ID, OKID)\) in \( OL \). If it exists, then returns \( OKID \) to \( A \), Otherwise, generates \( OKID \) and returns it to \( A \).

**Challenge:** A sends an identity \( ID^* \) and two same length messages \( M_0, M_1 \) to the challenger, in Phase 1, \( ID^* \) has not been executed the private key query or outsourcing key query. The challenger randomly chooses a bit \( \beta \in \{0, 1\} \) and forms the ciphertext \( CT^* = \text{Encrypt}(PK, M_\beta, ID^*) \), and then returns \( CT^* \) to \( A \).

**Phase 2:** The adversary \( A \) continues to issue private key queries and the challenger responds queries adaptively. However, \( ID^* \) will not been executed the private key query or outsourcing key query.

**Guess:** A outputs guess \( \beta' \in \{0, 1\} \).

In the above two games, we define the advantage of \( A \) as \( |Pr[\beta' = \beta] - (1/2)| \).
Different from Definition 1, Definition 2 adds the outsourcing private key generation algorithm, and it mainly adds the outsourcing key query in Phase 1. Moreover, in the above two security models, ID* will not be executed the private key query or outsourcing key query.

### 3. IDENTITY-BASED ENCRYPTION WITH OUTSOURCED PRIVATE KEY GENERATION

This section proposes an algorithm with verifiable outsourced private key generation based on the LW — IBE scheme [20], PKG can verify the correctness of the returned results effectively, but the server cannot obtain the PKG or the private key of any user.

#### 3.1 LW — IBE Scheme

In this section, we first review the LW — IBE scheme [20].

**Setup**($\lambda$): PKG runs the Setup algorithm, which selects a bilinear group $G$ of order $N = p_1p_2p_3$, where $p_1, p_2, p_3$ are three large primes. Let $G_{pi}$ be a subgroup of order $p_i$ in $G$, where $i \in \{1, 2, 3\}$. Then PKG randomly chooses $g, u, h \in G_{pi}$, $\alpha \in Z_N$, and generates a public key $PK = \{N, g, u, h, e(g, g)\}^\alpha$, and the master key is $\alpha \in Z_N$ and a generator of $G_{pi}$.

**KeyGen (MK, ID, PK):** PKG randomly selects $r_{ID} \in Z_N$, $R_{11}, R_{12} \in G_{p_3}$ based on identity ID, and computes:

$$K_1 = g^{\alpha R_{11}}, \quad K_2 = g^\alpha u^{ID} h^{ID} R_{12}.$$  

Then PKG outputs $SK_{ID} = (K_1, K_2)$ corresponding to ID.

**Encrypt** ($M, ID, PK$): For an identity ID and a message $M \in G_T$, we randomly choose $s \in Z_N$ and computes:

$$C_0 = Me(g, g)^{\alpha s}, \quad C_1 = (u^{ID} h)^s, \quad C_2 = g^s.$$  

Then creates the ciphertext $CT = (C_0, C_1, C_2)$.

**Decrypt** ($CT, SK_{ID}, PK$): Take as input an identity ID, a private $SK_{ID}$ and a ciphertext $CT$, then it decrypts the ciphertext as below:

$$e(C_1, K_1) = e(C_2, K_2)$$  

$$= Me(g, g)^{\alpha s} e((u^{ID} h)^s, g^{ID} R_{11})$$  

$$= Me(g, g)^{\alpha s} e(g^s, g^\alpha (u^{ID} h)^{ID} R_{12})$$  

$$= Me(g, g)^{\alpha s} e(g^s, g^\alpha)$$  

$$= M.$$  

Since $(u^{ID} h)^s, g^s \in G_{p_1}, R_{11}, R_{12} \in G_{p_3}$, then

$$e((u^{ID} h)^s, g^{ID} R_{11}) = e(g^s, R_{12}) = 1.$$  

Thus, the ciphertext can be decrypted successfully.

#### 3.2 The IBE Scheme with Verifiable Outsourced Key Generation

In this section, we propose an IBE scheme with verifiable outsourced key generation. The Setup, KeyGen, Encrypt, and Decrypt algorithms are run as the LW — IBE scheme, and then we only introduce the identity-based encryption with verifiable outsourced key generation. In [7], a subroutine Rand is used to generate random tuple with the form of $(c, c^{-1}, g^s \mod p)$, where $c \in Z_q$, so that the computations of the outsourcer can be speeded, and $U$ represents the server.

**KeyGenOut (ID):** (1) Firstly PKG computes $g^s$ and stores it in the system, then running Rand sixth to create six blinding pairs:

$$(t_1, t_1^{-1}, g^{t_1}), \quad (t_2, t_2^{-1}, g^{t_2}), \quad (t_3, t_3^{-1}, g^{t_3}), \quad (t_4, t_4^{-1}, g^{t_4}), \quad (t_5, t_5^{-1}, g^{t_5}), \quad (t_6, t_6^{-1}, g^{t_6}).$$

We denote $v_1 = g^{t_1 \mod p}, \quad v_2 = g^{t_2 \mod p}$, so that we get the first logical divisions:

$$(u^{ID} h)^{ID} = u^{ID} h^{ID}$$  

$$= (v_1 m)^{ID} (v_2 n)^{ID}$$  

$$= v_1^{ID} v_2^{ID} m^{ID} n^{ID}$$  

$$= g^{ID} g^{ID} g^{ID} g^{ID} g^{ID} g^{ID},$$

where $m = u/v_1 \mod p, \quad n = h/v_2 \mod p$.

Then we get the second logical divisions:

$$g^{ID} g^{ID} g^{ID} g^{ID} m^{ID} n^{ID} = g^s g^s g^s g^s m^{ID} n^{ID},$$

where

$$r_1 = ID^{ID} t_1 - t_3 \mod q,$$

$$r_2 = ID^{ID} t_2 - t_4 \mod q.$$

(2) PKG first selects $t$ random numbers $c_1, c_2, \ldots, c_t$, then randomly chooses $k \in [1, t], \quad x_1$ is a small
integer [from 1 to 10] and computes:

\[ d_1 = ID_{rID} - \sum_{j=1}^{k} c_j - x_1, \]
\[ d_2 = ID_{rID} - \sum_{j=k+1}^{t} c_j, \]

Then we set a collection \( A = \{c_1, c_2, \ldots, c_t, d_1, d_2\} \), and use a random permutation function to do the following \( A \to A_1 = \{g_1, g_2, \ldots, g_t\} \).

(3) Then PKG selects \( t \) random numbers \( e_1, e_2, \ldots, e_t \), then randomly chooses \( k \in [1, t] \), \( x_2 \) is a small integer [from 1 to 10] and computes:

\[ d_3 = rID - \sum_{j=1}^{k} e_j - x_2, \]
\[ d_4 = rID - \sum_{j=k+1}^{t} e_j, \]

Then we set a collection \( B = \{e_1, e_2, \ldots, e_t, d_3, d_4\} \), and use a random permutation function to do the following \( B \to B_1 = \{l_1, l_2, \ldots, l_{t+1}\} \).

Then we set \( OK_{ID} = (r_1, r_2, x_1, x_2, A_1, B_1, k, m, n) \).

(4) PKG makes queries to \( U \) in random order:

\[ U(g_i, m) \to m^g_i, \quad i \in [1, t+2], \]
\[ U(l_i, n) \to n^l_i, \quad i \in [1, t+2], \]
\[ U(r_1, t_5, g^5) \to g^{r_1}, \]
\[ U(r_2, t_9, g^9) \to g^{r_2}, \]
\[ U(x_1, m) \to m^{x_1}, \]
\[ U(x_2, n) \to n^{x_2}. \]

(5) PKG verifies that the calculation is correct or not:

(a) If

\[ g^{-1}(i) = d_1, \text{ then } s_1 = m^{d_1}, \]
\[ g^{-1}(i) = d_2, \text{ then } s_2 = m^{d_2}, \]
\[ g^{-1}(i) \in \{c_1, c_2, \ldots, c_t\}, \text{ then } s_3 = s_3 * m^{e_i}, \]
\[ g^{-1}(i) \in \{c_{k+1}, c_{k+2}, \ldots, c_t\} \text{ then } s_4 = s_4 * m^{e_i}, \]

where \( s_3 \) and \( s_4 \) initial value are 1.

(b) If

\[ l^{-1}(i) = d_3, \text{ then } s_5 = n^{d_3}, \]
\[ l^{-1}(i) = d_4, \text{ then } s_6 = n^{d_4}, \]
\[ l^{-1}(i) \in \{e_1, e_2, \ldots, e_t\}, \text{ then } s_7 = s_7 * n^{e_i}, \]
\[ l^{-1}(i) \in \{e_{k+1}, e_{k+2}, \ldots, e_t\}, \text{ then } s_8 = s_8 * n^{e_i}, \]

where \( s_7 \) and \( s_8 \) initial value are 1.

The verification calculation is as follows:

\[ m^{d_1} \cdot \prod_{j=1}^{k} m_j \cdot m^{e_1} = m^{d_2} \cdot \prod_{j=k+1}^{t} m_j, \]
\[ n^{d_3} \cdot \prod_{j=1}^{k} n_j \cdot n^{e_2} = n^{d_4} \cdot \prod_{j=k+1}^{t} n_j, \]

If not, PKG outputs “error”; otherwise, PKG randomly chooses \( R'_{21}, R'_{22} \in G_p \), and computes:

\[ K_1 = g^{\alpha_{ID} R'_{21}}, \]
\[ K_2 = g^\alpha g^{\alpha_{ID}} g^{\alpha_{ID}} g^{\alpha_{ID}} m^{ID_{rID} h^{ID} R'_{22}} = g^\alpha u^{ID_{rID} h^{ID} R'_{22}} = g^\alpha (u^{ID} h)^{ID} R'_{22}. \]

The private key corresponding to the \( ID \) is \( SK_{ID} = (K_1, K_2) \).

In the above outsourcing algorithm, the server need not know the relationship between \( OK_{ID} \) and \( ID \), and users only need to perform modular exponentiation operations once in the outsourcing process, which greatly reduces the computational cost of PKG, and the server cannot obtain the PKG or the private key of any user.

4. ANALYSIS OF THE SCHEME

This section analyzes the IBE – VOK scheme, including the indistinguishability of the ciphertext and verifiability of the outsourcing results. Then, we show the computation cost comparison of PKG for generating private keys directly and outsourcing the task to a single server, and an efficiency comparison for different schemes with outsourced key generation. Finally, the simulation of the scheme is implemented.

4.1 Security Analysis

When the proposed scheme is under the choice of plain-text attack (CPA), Theorem 1 and Theorem 2 prove the
indistinguishability of the ciphertext and verifiability of the outsourcing results, respectively.

**Theorem 1:** (Indistinguishability of Ciphertext):
Assuming that the LW − IBE scheme [20] is completely secure under CPA attack, then the IBE − VOK scheme is also completely secure under CPA attack.

**Proof:** Suppose that an adversary $A$ can distinguish two different plaintexts with a probability of $\varepsilon$ under the CPA security model of the IBE − VOK scheme, then the algorithm $B$ can be constructed to distinguish two different plaintexts with the same probability of $\varepsilon$ under the CPA security model of the LW − IBE scheme.

Let $C$ represent the challenger who answered $B$ in the LW − IBE scheme, $A$ and $B$ perform the following steps:

**Setup:** $C$ sends $PK = \{N, g, u, h, e(g, g)^{a}\}$ to $B$ as a public parameter of the LW − IBE scheme, and $B$ returns $PK$ to $A$ as a public parameter of the IBE − VOK scheme.

**Phase 1:** Firstly, $B$ creates an empty outsourced list $OL$ and then performs the following inquiries.

Private key query: $A$ sends identity $ID$ to $B$, then $B$ sends the ID to $C$ for private key query in the LW − IBE scheme and $C$ generates $SK_{ID} = (K_1, K_2)$, where $K_1 = g^{rID} R_{11}$, $K_2 = g^{uID} h^{rID} R_{12}$, which is returned to $B$ and $B$ returns $SK_{ID}$ to $A$.

Outsourcing key query: $A$ sends identity $ID$ to $B$, $B$ searches $(ID, OK_{ID})$ in $OL$ and returns $OK_{ID}$ to $A$ if it exists, otherwise randomly selects $r_{ID}, a \in Z_N$, and calculates $OK_{ID} = (r_{ID}, r_{ID}, x_1, x_2, A_1, B_1, k, m, n)$, then adds the $(ID, OK_{ID})$ to the outsourced list $OL$ and returns $OK_{ID}$ to $A$.

**Challenge:** $A$ sends two plaintexts $M_0, M_1$ of the same length and an identity $ID^*$ to $B$, where $ID^*$ does not perform the outsourcing key query or private key query. Then $B$ sends them to $C$ to complete the challenge phase of the LW − IBE scheme. $C$ randomly chooses $\beta \in \{0, 1\}$, then generates ciphertext $CT^* = (C_0^*, C_1^*, C_2^*)$ of $M_\beta$, and returns $CT^*$ to $B$. At last, $B$ sends $CT^*$ to $A$ as the challenge ciphertext.

**Phase 2:** $A$ continues to ask, and $B$ runs the same operation as Phase 1. However, $ID^*$ can not be executed the outsourcing key query or private key query.

**Guess:** $A$ outputs $\beta' \in \{0, 1\}$, $B$ also outputs $\beta'$ as a guess for $\beta$.

Then we analyze $B$’s success probability. When $A$ runs the outsourcing key query about identity $ID$, the value of $r_1, r_2, x_1, x_2, A_1, B_1, k, m, n$ can be obtained by $ID$ and $OK_{ID}$. If $A$ continues to ask $ID$ for private key query and it can obtain $SK_{ID}$. Therefore, if $A$ can break the IBE − VOK scheme with a probability of $\varepsilon$, then $B$ can also break the LW − IBE scheme with the same probability of $\varepsilon$.

In the above game, $B$ successfully simulates the full security model under the CPA attack of the IBE − VOK scheme. As in [20], the LW − IBE scheme is completely secure under CPA attack, similarly, the IBE − VOK scheme is also completely secure under CPA attack according to theorem 1. Then the verifiability of the IBE − VOK scheme is proved as below.

**Theorem 2:** (Verifiability of outsourcing results): In a malicious model, PKG can detect the error with a probability of $(2t + 6)/(2t + 8)$ if the server returns incorrect results in the verifiable IBE scheme, $(t$ represents the number of random numbers).

**Proof:** Assume $U$ is a malicious server, at the end of the algorithm, the PKG verifies the result as below:

\[
m^{d_1} \ast \prod_{j=1}^{k} m^{\gamma_j} \ast m^{x_1} = m^{d_2} \ast \prod_{j=k+1}^{t} m^{\gamma_j},
\]

\[
n^{d_3} \ast \prod_{j=1}^{k} n^{\nu_j} \ast n^{x_2} = n^{d_4} \ast \prod_{j=k+1}^{t} n^{\nu_j},
\]

Since the PKG sends $(r_1, r_2, x_1, x_2, A_1, B_1, k, m, n)$ to the server, and $U$ must return true values of $m^{\theta}, n^{\theta}, g^{\nu_1}, g^{\nu_2}, m^{\nu_1}, n^{\nu_2}$. Otherwise, the Equations (1) and (2) cannot pass the verification successfully.

Therefore, if $U$ returns the error result, the PKG will surely detect this error with a probability of $(2t + 6)/(2t + 8)$ in the proposed IBE − VOK scheme, $(t$ is the number of random numbers).

### 4.2 Efficiency Comparison

Suppose the system has a total of $m$ users. PKG can pre-compute $g^a$ and store it in the system. The following is a computational comparison of PKG between generating private keys directly and outsourcing the task to a single server, as shown in Table 1.

In Tables 1 and 2, $m$ represents the total number of system users and $n$ is the bit length of $N$. Note that $G$ and $G_T$ are
cyclic groups of order $N = p_1 p_2 p_3$, where $p_1, p_2, p_3$ are all 512-bit primes. MExp, MM denote the computation of modular exponentiation and modular multiplication, $t$ is the number of random numbers.

As we know, using the table-lookup method to invoke the Rand subroutine takes $O(1)$ MM, and using the square-and-multiply method to execute the modular exponentiation operation takes $1.5n$ MM. In the outsourcing algorithms, we omit other operations such as modular addition.

It can be seen from Table 1, if PKG directly generates the private keys for the users and need to perform $3m + 1$ MExp, $4m$ MM operations and the total computation cost is roughly $4.5mn$ MM. Then if PKG outsources the task of generating private keys to a single server, it only needs to execute 1 MExp, $2tm + 15m$ MM and $6m$ Rand operations for $m$ users, the total computation cost is roughly $1.5n$ MM. The server $U$ runs $2tm + 8m$ MExp operations. Therefore, the cost of the PKG is much smaller than that of generating private keys directly.

As shown in Table 2, the proposed scheme is based on a single server compared with [19], and PKG only needs to perform modular exponentiation operations once during the outsourcing process. Moreover, with the increasing number of users in the system, the computation of PKG remains basically unchanged, and the total computation cost is roughly $1.5n$ MM in this paper. However, in scheme [19], the total computation cost is roughly $1.5mn$ MM and it is much more expensive than this scheme, so the scheme of this paper has more practical advantages.

### Table 1: Computation cost comparison of PKG for generating keys directly and outsourcing the task to a single server

| Algorithm       | PKG generates key directly | Outsourcing calculation | PKG | U |
|-----------------|---------------------------|-------------------------|-----|---|
| MExp            | $3m + 1$                  | 1                       | $2tm + 8m$ |
| MM              | $4m$                      | $2tm + 15m$             | 0   | 0 |
| Invoke(Rand)    | 0                         | $6m$                    | 0   | 0 |
| Total(MM)       | $\approx 4.5mn$          | $\approx 1.5n$         | $3tm + 12mn$ |

### Table 2: Computation cost comparison for different schemes with outsourced key generation

| PKG calculation | Scheme [19] | This paper |
|-----------------|-------------|------------|
| Server          | 2           | 1          |
| MExp            | $m+1$       | 1          |
| MM              | $7m$        | $2tm + 15m$|
| Invoke(Rand)    | $4m$        | $6m$       |
| Total(MM)       | $\approx 1.5mn$ | $\approx 1.5n$ |

### Figure 2: Simulation for the proposed IBE scheme with outsourced key generation

#### 4.3 Simulation Result

In this section, we provide the experiment evaluation to show the efficiency of the proposed algorithm based on one untrusted server. The PKG and the cloud server are simulated by Intel Core i5 Processor running at 3.2 GHz with 8G memory and Intel Xeon Processor running at 2.2, 2.19 GHz (two processors) with 128G memory, respectively. Note that $G$ and $G_T$ are cyclic groups of order $N = p_1 p_2 p_3$, where $p_1, p_2, p_3$ are all 512-bit primes. The programming language is Java.

In Figure 2, we give the time cost of PKG for generating private keys directly and outsourcing the task to the server of the IBE scheme respectively. In the figure, the abscissa represents the total number of users in the system and the ordinate represents the total time cost. It is obvious that the time cost for PKG and the server are less than half of generating private keys directly, with the increasing number of users in the system, the total calculation of the server and PKG is also less than generating private keys directly. For example, when the number of system users is $2^{11}$, the PKG calculation time is about 1100 s and the server is about 1400 s, however, the time cost of direct computation is about 4700 s, obviously, it is far higher than the sum of PKG and the server. Therefore, the algorithm of this paper will become more practical when the number of users is continuously increasing.

### 5. CONCLUSION

In this paper, we use combinatorial bilinear groups to propose a verifiable outsourcing algorithm for private
key generation based on identity-based encryption. The scheme greatly reduces the time cost of PKG, and the server cannot obtain PKG or the private keys of any user during the outsourcing process. PKG outsources the tasks to a single server, and it can improve security and verify the correctness of outsourced results. Finally, the experimental results show that the total calculation of the server and PKG is far less than generating private keys directly.

**FUNDING**

This work was supported by the National Natural Science Foundation of China [grant numbers U1736120, 61572309, U1536109], and open project of Shanghai Key Laboratory of Scalable Computing and Systems.

**REFERENCES**

1. J. Ye, X. Chen, and J. Ma, “An improved algorithm for secure outsourcing of modular exponentiations,” in IEEE International Conference on Advanced Information Networking and Applications Workshops, 2015, pp. 73–76.

2. Y. Ren, M. Dong, Z. Qian, X. Zhang, and G. Feng, “Efficient algorithm for secure outsourcing of modular exponentiation with single server,” IEEE Transactions on Cloud Computing. doi:10.1109/TCC.2018.2851245

3. L. Zhang, and S. R. Naini, “Private outsourcing of polynomial evaluation and matrix multiplication using multilinear maps,” in Cryptology and Network and Security [C], 2013, pp. 329–48.

4. H. Ma, R. Zhang, Z. Wan, Y. Lu, and S. Lin, “Verifiable and excusable outsourced attribute-based encryption for access control in cloud computing,” IEEE Transactions on Dependable and Secure Computing, Vol. 14, pp. 461–62, 2017.

5. J. Li, Y. Wang, Y. Zhang, and J. Han. “Full verifiability for outsourced decryption in attribute based encryption,” in IEEE Transactions on Services Computing, 2017, pp. 1–1.

6. Y. Ren, X. Zhang, G. Feng, Z. Qian, and F. Li, “How to extract image features based on co-occurrence matrix securely and efficiently in cloud computing,” IEEE Transactions on Cloud Computing. doi:10.1109/TCC.2017.273980

7. S. Hohenberger, and A. Lysyanskaya. “How to securely outsource cryptographic computations,” in TCC. LNCS, Vol. 3378, J. Kilian, Ed. Heidelberg: Springer, 2005, pp. 264–82.

8. Y. Wang, Q. Wu, and D. Wong. “Securely outsourcing exponentiations with single untrusted program for cloud storage,” ESORICS, Part I, LNCS 8712, Springer Switzerland, pp. 326–43, 2014.

9. J. Liu, and B. Yang. “A new algorithm for secure outsourcing composite modular exponentiation,” in: IEEE 2nd International Conference on Information Science and Control Engineering, 2015, pp. 461–65.

10. J. Ye, and J. Wang. “Secure outsourcing of modular exponentiation with single untrusted server,” in: 18th International Conference on Network-Based Information Systems, 2015, pp. 643–45.

11. X. Chen, W. Susilo, J. Li, D. S. Wong, J. Ma, S. Tang, and Q. Tang, “Efficient algorithms for secure outsourcing of bilinear pairing,” Theoretical Computer Science, Vol. 562, pp. 112–21, 2015.

12. H. Tian, F. Zhang, and K. Ren, “Secure bilinear pairing outsourcing made more efficient and flexible,” in: Proceedings of ASIA CCS, Singapore, April 14-17, ACM Press, New York, 2015, pp. 417–26.

13. Y. Ren, M. Dong, Z. Niu, and X. Du, “Noninteractive verifiable outsourcing algorithm for bilinear pairing with improved checkability,” Security and Communication Networks, 2017, pp. 1–9, 2017.

14. A. Shamir, “Identity-based cryptosystems and signature schemes,” In Proc. CRYPTO 84, California, USA, Aug. 19–22. Springer, Berlin, 1984, pp. 47–53.

15. D. Boneh, and M. Franklin, “Identity-based encryption from the weil pairing,” Proc. CRYPTO 01, California, USA, Aug. 19-23. Springer, Berlin, 2001, pp. 213–29.

16. Y. Ren, and D. Gu, “Cca2 secure (hierarchical) identity-based parallel key-insulated encryption without random oracles,” Journal of Systems and Software, Vol. 83, pp. 153–62, 2010.

17. J. Yu, R. Hao, H. Zhao, M. Shu, and J. Fan, “Iribe: intrusion-resilient identity-based encryption,” Information Sciences, Vol. 329, pp. 90–104, 2016.

18. J. Li, X. Chen, C. Jia, and W. Lou, “Identity-based encryption with outsourced revocation in cloud computing,” IEEE Transactions on Computers, 64, pp. 425–37, 2015.

19. Y. Ren, and J. Cai, “Verifiable outsourcing private key generation algorithm in an identity-based encryption scheme,” Journal of Communication (in Chinese), Vol. 36, pp. 1–6, 2015.

20. A. Lewko, and B. Waters, “New techniques for dual system encryption and fully secure HIBE with short ciphertexts,” Theory of Cryptography Conference [C], pp. 455–79, 2010.
Authors

**Ting Xue** received the BS degree on electrical and information engineering from Changshu Institute of Technology, China, in 2016. Currently, she is a graduate student at the School of Communication and Information Engineering in Shanghai University, Shanghai, China. Her research interests include applied cryptography and secure outsourcing computing.

**Email:** xueting1505@163.com

**Yanli Ren** is an associate professor in School of Communication and Information Engineering at Shanghai University, China. She was awarded an MS degree in applied mathematics in 2005 from Shaanxi Normal University, China, and a PhD degree in computer science and technology in 2009 from Shanghai Jiao Tong University, China. Her research interests include applied cryptography, secure outsourcing computing, and network security.

**Corresponding author. Email:** renyanli@shu.edu.cn

**Guorui Feng** received the BS and MS degree in computational mathematic from Jilin University, China, in 1998 and 2001 respectively. He received PhD degree in electronic engineering from Shanghai Jiaotong University, China, 2005. From January 2006 to December 2006, he was an assistant professor in East China Normal University, China. During 2007, he was a research fellow in Nanyang Technological University, Singapore. Now, he is with the School of Communication and Information Engineering, Shanghai University, China. His current research interests include image processing, image analysis, and computational intelligence.

**Email:** grfeng@shu.edu.cn