Soft Comput
DOI 10.1007/s00500-016-2271-2

METHODOLOGIES AND APPLICATION

Verifiable outsourced ciphertext-policy attribute-based encryption in cloud computing

Hao Wang[1] · **Debiao He**[2] · **Jian Shen**[3] · **Zhihua Zheng**[1] · **Chuan Zhao**[4] · **Minghao Zhao**[4]

Abstract In the attribute-based encryption (ABE) systems, users can encrypt and decrypt messages based on their attributes. Because of the flexibility of ABE, it is more and more widely used in various network environments. However, complex functionality of ABE may cause an enormous computational cost. This reason greatly restricts the application of ABE in practice. In order to minimize the local computation of ABE, we introduce the concept of verifiable outsourced ABE system, in which key generation center, encryptor and decryptor, are able to outsource their computing tasks to the corresponding service providers, respectively, to reduce the local load. In addition, they are also able to verify the correctness of outsourcing calculation efficiently by using the outsourcing verification services. This is useful to save local computational resources, especially for mobile devices. Then, we propose a specific verifiable outsourced ABE scheme and prove its adaptive security in the standard model using the dual-system encryption method. Finally, we introduce how to deploy our outsourced CP-ABE scheme in cloud computing environment.

Keywords CP-ABE · Outsourcing · Verifiable · Cloud computing · Adaptively secure

Communicated by V. Loia.

✉ Debiao He
hedebiao@163.com

Hao Wang
wanghao@sdnu.edu.cn

Jian Shen
jianshen_nuist@sina.com

Zhihua Zheng
zhengzh121@sina.com

Chuan Zhao
zhaochuan.sdu@gmail.com

Minghao Zhao
zhaominghao@hrbeu.edu.cn

1 School of Information Science and Engineering, Shandong Normal University, Jinan, China

2 State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China

3 School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China

4 School of Computer Science and Technology, Shandong University, Jinan, China

1 Introduction

In 2005, Sahai and Waters (2005) first introduced the concept of attribute-based encryption (ABE). It is an extension of identity-based encryption in which the "identity" of users are dependent upon their attributes (e.g., Shandong Normal University, male, teacher). In such a system, the decryption is correct only if the set of attributes of the user key matches the access policy of the ciphertext. Then, Goyal et al. (2006) give the formal definition of ABE system, and divide this concept into two types: ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). In CP-ABE, ciphertexts are associated with access policies and keys are associated with sets of attributes, while in KP-ABE, keys are associated with access policies and ciphertexts are associated with sets of attributes.

In 2007, the first CP-ABE scheme was proposed in the random oracle model by Bethencourt et al. (2007). Then Cheung and Newport (2007) proposed an "AND-gate" CP-ABE scheme without random oracle (the access policies only support "AND-gate"). In 2008, Waters (2008) proposed the first expressive CP-ABE using linear secret-sharing schemes

Published online: 13 July 2016

(LSSS) as access policies and proved security in the selective model. Until 2010, Lewko proposed the first fully secure ABE schemes in Lewko et al. (2010), using the dual-system encryption method. In the current ABE systems, access policies are typically expressed by LSSS, which makes keys and ciphertexts have rich structure.

1.1 Motivation and contribution

The main challenge in employing ABE systems in the real network environment is that the complex functionality of ABE may cause an enormous computational cost. This large expense may impact several networking applications, especially for mobile networks.

In this paper, we aim to minimize the local computation for CP-ABE and introduce a new method to construct the outsourced CP-ABE. Our outsourced CP-ABE includes four types of service providers for outsourcing tasks, i.e., outsourcing key generation server provider (OKGSP), outsourcing encryption server provider (OESP), outsourcing decryption server provider (ODSP) and outsourcing verification server provider (OVSP). To improve the computational efficiency, KGC can outsource part of the calculation tasks of key generation to OKGSP, encryptor can outsource part of the calculation tasks of encryption to OESP and decryptor can outsource part of the calculation tasks of decryption to ODSP. In addition, all entities can verify the correctness of the outsourcing calculation by calling the outsourcing verification server provider. It is useful for mobile devices to save local resources. Then, we proposed a specific outsourced CP-ABE scheme and proved its security in the standard model. In our new outsourced CP-ABE scheme, the local computation of each stage is constant, and users can verify the correctness of the outsourcing results efficiently. Finally, we introduced how to deploy our outsourced CP-ABE scheme in cloud computing environment and introduced a specific application in secure cloud storage system.

1.2 Related work

To reduce the local computation cost, the natural idea is to deliver most of computational tasks outside. In actual fact, theoretical computer science community has paid attention to the problem that how to outsource different kinds of expensive computations securely (Atallah et al. 2001; Atallah and Li 2005; Benjamin and Atallah 2008; Chen et al. 2012). Recently, Chen et al. presented efficient algorithms for secure outsourcing of modular exponentiations (Chen et al. 2014), linear equations (Chen et al. 2015a) and bilinear pairings (Chen et al. 2015b). However, we note that though kinds of expensive computations can be securely outsourced, they are difficult to be used directly to construct the provably secure outsourced CP-ABE.

To reduce the local load of ABE, Green et al. (2011) provided a novel method for outsourcing the decryption of ABE, while Li et al. (2012) presented a novel ABE scheme, which could outsource both encrypting computation and decrypting computation to MapReduce cloud securely. Then, Lai et al. (2013) proposed a verifiable outsourced ABE scheme, which allows to verify the correctness of decryption. Their scheme appends additional information with the ciphertexts and uses this information for verification. In 2014, Li et al. (2014) presented a new outsourced ABE scheme, which allows to outsource key generating computation and enables checkability on returned results from cloud. Recently, Wang et al. (2015) proposed an adaptively secure outsourced CP-ABE scheme in the standard model; however, they only considered the outsourcing needs of the decryption and did not verify the outsourcing results.

Furthermore, because of the flexibility of ABE, it is more and more widely used in various network environments, such as cloud computing (Zhangjie et al. 2015b; Ren et al. 2015; Zhangjie et al. 2015a), social networking (Shen et al. 2015a), web of things (Shen et al. 2015b; He et al. 2015; He et al. 2016b), access control system (He et al. 2016a), authentication system (Huang et al. 2014; Huang et al. 2015), and so on.

1.3 Organization

We introduce the preliminaries in Sect. 2 and provide the relevant definitions for outsourced ABE in Sect. 3. In Sect. 4 and Sect. 5, we construct a specific outsourced CP-ABE scheme and give the security proof, respectively. Then, we introduce how to deploy our outsourced CP-ABE scheme in cloud computing environment in Sect. 6. Finally, we give the conclusion in Sect. 7.

2 Preliminaries

2.1 Composite order bilinear group

The concept of composite order bilinear group was first introduced by Boneh et al. (2005).

$\mathcal{G}(\lambda) \rightarrow (N, G, G_T, e)$: $\mathcal{G}$ is a group generation algorithm. It takes a security parameter λ as input and outputs a description of bilinear cyclic group with order N, where $N = \prod_1^n p_i$ is a product of n distinct primes, and $e : G^2 \rightarrow G_T$ is map such that:

- (Bilinear) $\forall g, h \in G, a, b \in Z_N, e(g^a, h^b) = e(g, h)^{ab}$.
- (Non-degenerate) $\exists g \in G$ such that $e(g, g)$ has order N in G_T.
- (Computable) The group operations in G and G_T and the map e are computable in polynomial time with respect to λ.

– (Orthogonal) If $f_i \in G_{p_i}$ and $f_j \in G_{p_j}$ for $i \neq j$, then $e(f_i, f_j)$ is the identity element in G_T.

In this paper, we set $n = 3$, i.e., $N = p_1 p_2 p_3$. For any non-empty set $Z \subseteq \{1, 2, 3\}$, there is a corresponding subgroup of G of order $\prod_{i \in Z} p_i$. Let G_Z denote this subgroup.

2.2 Complexity assumption

Assumption 1 Lewko and Waters (2012) Let $Z_0, Z_1, Z_2, \ldots, Z_k$ denote a collection of non-empty subsets of $\{1, 2, 3\}$ where each Z_i for $i \geq 2$ satisfies either $Z_0 \bigcap Z_i \neq \varnothing \neq Z_1 \bigcap Z_i$ or $Z_0 \bigcap Z_i = \varnothing = Z_1 \bigcap Z_i$. We define the following distribution:

$$\mathbf{G} := (N = p_1 p_2 p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$
$$g_{Z_2} \xleftarrow{R} G_{Z_2}, \ldots, g_{Z_k} \xleftarrow{R} G_{Z_k},$$
$$D := (\mathbf{G}, g_{Z_2}, \ldots, g_{Z_k}),$$
$$T_0 \xleftarrow{R} G_{Z_0}, T_1 \xleftarrow{R} G_{Z_1}$$

Fixing the collection of sets $Z_0, \ldots, Z_k$, the advantage of an algorithm $\mathcal{A}$ in breaking this assumption is defined to be:

$$\mathrm{Adv}^1_{\mathcal{G},\mathcal{A}}(\lambda) := |Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1]|.$$

If $\mathrm{Adv}^1_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function, we say that $\mathcal{G}$ satisfies Assumption 1.

Assumption 2 Lewko and Waters (2010), Lewko et al. (2010), Lewko and Waters (2012) We define the following distribution:

$$\mathbf{G} := (N = p_1 p_2 p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$
$$g_1 \xleftarrow{R} G_{p_1}, g_2, X_2, Y_2 \xleftarrow{R} G_{p_2}, g_3 \xleftarrow{R} G_{p_3},$$
$$\alpha, s \xleftarrow{R} \mathbb{Z}_N$$
$$D := (\mathbf{G}, g_1, g_2, g_3, g_1^{\alpha} X_2, g_1^{s} Y_2),$$
$$T_0 = e(g_1, g_1)^{\alpha s}, T_1 \xleftarrow{R} G_T$$

The advantage of an algorithm $\mathcal{A}$ in breaking this assumption is defined to be:

$$\mathrm{Adv}^2_{\mathcal{G},\mathcal{A}}(\lambda) := |Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1]|.$$

If $\mathrm{Adv}^2_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function, we say that $\mathcal{G}$ satisfies Assumption 2.

3 Definition of outsourced ABE

3.1 Algorithms

In an outsourced ABE system, there are seven types of entities, namely key generation center (KGC), encryptor, decryptor, outsourcing key generation server provider (OKGSP), outsourcing encryption service provider (OESP), outsourcing decryption service provider (ODSP), outsourcing decryption service provider (ODSP), and outsourcing verification service provider (OVSP). They interact as in Fig. 1 and run the following seven algorithms:

– **Setup**$(\lambda, \mathcal{U}) \rightarrow PP, MSK$ The setup algorithm takes security parameter λ and attribute universe $\mathcal{U}$ as input and outputs public parameters PP and master secret key MSK. PP may be widely distributed, while MSK is known only to key generation center (KGC).
– **Outsourcing-KeyGen-PreProc**$(PP, MSK) \rightarrow MSK_O, MSK_L$ The outsourcing key generation preprocessing algorithm is run by key generation center (KGC); it takes the public parameters PP and the master secret key MSK as input. It outputs an outsourcing master secret key MSK_O and a local master secret key MSK_L. Then, it delegates the MSK_O to its outsourcing key generation server provider (OKGSP) and keeps MSK_L locally.
– **Outsourcing-KeyGen**$(PP, MSK_O, I_{key}) \rightarrow IK_{I_{key}}$ The outsourcing key generation algorithm is run by outsourcing key generation service provider (OKGSP); it takes the PP, MSK_O, and an attribute set (resp. access structure) I_{key} as input. It outputs an intermediate key $IK_{I_{key}}$ and sends $IK_{I_{key}}$ to KGC.
– **KeyGen**$(PP, MSK_L, IK_{I_{key}}) \rightarrow SK_{I_{key}}$ The key generation algorithm is run by key generation center (KGC); it takes the public parameters PP, the local master secret key MSK_L, the intermediate key $IK_{I_{key}}$ as input, and outputs a user secret key $SK_{I_{key}}$ for attribute set (resp. access structure) I_{key}.

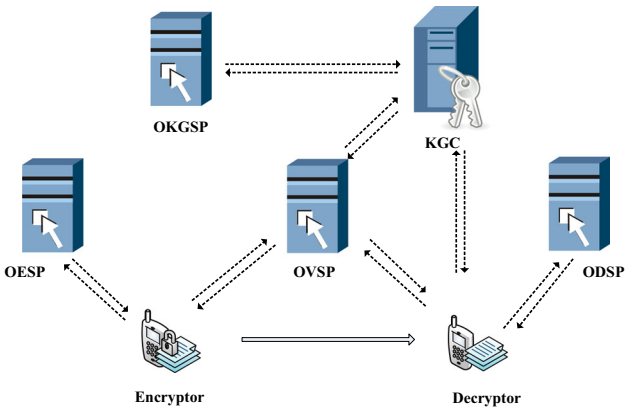

Fig. 1 Outsourced ABE System

- **Outsourcing-Encryption-PreProc**$(PP) \rightarrow EP_O, EP_L$ The outsourcing encryption preprocessing algorithm is run by encryptor; it takes as input the public parameters PP and outputs an outsourcing encryption parameters EP_O and a local encryption parameters EP_L. The encryptor keeps EP_L locally and delegates the EP_O to its outsourcing encryption service provider (OESP).
- **Outsourcing-Encryption**$(PP, EP_O, I_{enc}) \rightarrow IC_{I_{enc}}$ The outsourcing encryption algorithm is run by OESP; it takes as input the public parameters PP, the outsourcing encryption parameters EP_O, and an access structure (resp. attribute set) I_{enc}. It outputs an intermediate ciphertext $IC_{I_{enc}}$.
- **Encryption**$(PP, EP_L, IC_{I_{enc}}, m) \rightarrow CT_{I_{enc}}$ The encryption algorithm is run by encryptor; it takes as input the public parameters PP, the local encryption parameters EP_L, the intermediate ciphertext $IC_{I_{enc}}$ and a message m. It outputs the ciphertext $CT_{I_{enc}}$.
- **Outsourcing-Decryption-PreProc**$(SK) \rightarrow (SK_O, SK_L)$ The outsourcing decryption preprocessing algorithm is run by decryptor; it takes its secret key SK as input and outputs an outsourcing secret key SK_O and a local secret key SK_L. The decryption user keeps SK_L locally and delegates the SK_O to its outsourcing decryption service provider (ODSP).
- **Outsourcing-Decryption**$(SK_O, CT) \rightarrow CT'$ The outsourcing decryption algorithm is run by ODSP; it takes as input an outsourcing secret key SK_O for attribute set (resp. access structure) I_{key} and a ciphertext CT that was encrypted under access structure (resp. attribute set) I_{enc}. It outputs the partially decrypted ciphertext CT' if $f(I_{enc}, I_{key}) = 1$ and the error symbol $\perp$ otherwise.
- **Decryption**$(CT', SK_L) \rightarrow m$ The decryption algorithm is run by decryptor; it takes SK_L for I_{key} and CT' for I_{enc} as input. It outputs the original message m if $f(I_{enc}, I_{key}) = 1$ and the error symbol $\perp$ otherwise.
- **Outsourcing-Verification** The outsourcing verification algorithm is run by outsourcing verification service provider (OVSP) and provides verification services to confirm whether the outsourcing is correct. It contains two sub-algorithms:
  - **Outsourcing-KeyGen-Verification**$(PP, MSK_O, IK_{I_{key}}) \rightarrow b \in \{0, 1\}$ It takes the public parameters PP, an outsourcing master secret key MSK_O and a intermediate key $IK_{I_{key}}$ as input. It outputs 1 if $IK_{I_{key}}$ is correctly calculated by OKGSP using MSK_O and 0 otherwise.
  - **Outsourcing-Encryption-Verification**$(PP, EP_O, IC_{I_{enc}}) \rightarrow b \in \{0, 1\}$ It takes the public parameters PP, an outsourcing encryption parameters EP_O and a intermediate ciphertext $IC_{I_{enc}}$ as input. It outputs 1 if $IC_{I_{enc}}$ is correctly calculated by OESP using EP_O and 0 otherwise.

  Note, in decryption phase, the outsourcing verification service is not used, because the decryptor can verify the correctness of the outsourcing calculation by the plaintext m easily.

3.2 Security model

The security experiment OABE-Exp$_{\mathcal{A},\Pi}(\lambda, \mathcal{U})$ for outsourced ABE scheme Π, adversary $\mathcal{A}$, parameter λ and attribute universe $\mathcal{U}$ is defined as follow:

Initial The challenger runs the setup algorithm to obtain PP and MSK. Then, it runs the Outsourcing-KeyGen-PreProc algorithm on (PP, MSK) to obtain MSK_O and MSK_L. It gives PP and MSK_O to the adversary.

Note: In real-world situations, we allow the adversary to corrupt OKGSP, so that the adversary can obtain the MSK_O in this experiment.

Phase 1 The challenger sets two integer counters $\zeta, \xi = 0$, two empty tables $\mathcal{T}_1, \mathcal{T}_2$, and an empty set $\mathcal{D}$. The adversary can make the following queries repeatedly:

- *Create*(I_{key}) The challenger sets $\zeta := \zeta + 1$. It runs Outsourcing-KeyGen algorithm on I_{key} to obtain $IK_{I_{key}}$, runs KeyGen algorithm on $IK_{I_{key}}$ to obtain $SK_{I_{key}}$ and runs Outsourcing-Decryption-PreProc algorithm on $SK_{I_{key}}$ to obtain $SK_{I_{key},O}$ and $SK_{I_{key},L}$. Then, it stores in table $\mathcal{T}_1$ the entry $(\zeta, I_{key}, IK_{I_{key}}, SK_{I_{key}}, SK_{I_{key},O}, SK_{I_{key},L})$, and gives $(\zeta, I_{key}, IK_{I_{key}}, SK_{I_{key},O})$ to adversary.

  Note: In real-world situations, we allow the adversary to corrupt OKGSP and ODSP, so that the adversary can obtain the $IK_{I_{key}}$ and $SK_{I_{key},O}$ in this experiment.
- *Corrupt*(i) If the ith entry exists in table $\mathcal{T}_1$, the challenger gets the entry $(i, I_{key}, IK_{I_{key}}, SK_{I_{key}}, SK_{I_{key},O}, SK_{I_{key},L})$, sets $\mathcal{D} := \mathcal{D} \cup \{I_{key}\}$ and returns the key set $(i, SK_{I_{key}}, SK_{I_{key},L})$ to the adversary, else it returns $\perp$.
- *Outsourcing-Encryption*(I_{enc}) The challenger sets $\xi := \xi + 1$. It runs Outsourcing-Encryption-PreProc algorithm to obtain EP_O and EP_L and runs Outsourcing-Encryption algorithm on (EP_O, I_{enc}) to obtain $IC_{I_{enc}}$. Then, it stores in table $\mathcal{T}_2$ the entry $(\xi, I_{enc}, EP_O, EP_L, IC_{I_{enc}})$ and gives $(\xi, I_{enc}, EP_O, IC_{I_{enc}})$ to adversary.

  Note: In real-world situations, we allow the adversary to corrupt OESP, so that the adversary can obtain the EP_O and IC in this experiment.

Challenge The adversary provides two messages m_0^* and m_1^* with equal length, an index j of $\mathcal{T}_2$. If there exists a j^{th} entry $(j, I_{enc}^*, EP_O, EP_L, IC_{I_{enc}^*})$ in table $\mathcal{T}_2$, and for all $I_{key} \in \mathcal{D}$, $f(I_{enc}^*, I_{key}) \neq 1$, the challenger chooses a random bit b and encrypts the message m_b^* under $(I_{enc}^*, EP_L, IC_{I_{enc}^*})$. The

ciphertext CT^* is returned to the adversary. Otherwise, it returns $\perp$.

Phase 2 Just like Phase 1, except that the adversary cannot obtain the suitable keys for the challenge ciphertext. That is, the adversary cannot issue any *Corrupt* queries that would added I_{key} which satisfies $f(I^*_{enc}, I_{key}) = 1$ to $\mathcal{D}$.

Guess The adversary outputs a guess b' of b. The output of the experiment is 1 if and only if $b = b'$.

Definition 1 An outsourced ABE scheme is secure if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a negligible function $negl$ such that:

$$\Pr[\text{OABE-Exp}_{\mathcal{A},\Pi}(\lambda, \mathcal{U}) = 1] \leq \frac{1}{2} + negl(\lambda).$$

4 Outsourced CP-ABE scheme

4.1 Construction

In this section, we proposed a specific outsourced CP-ABE scheme. In our construction, we will use linear secret-sharing schemes (LSSS) to express the access structure. This technology is proposed by Beimel (1996) and widely used in ABE scheme, such as Waters (2011), Lewko and Waters (2012) etc. The proposed scheme consists of the following 12 algorithms:

Setup$(\lambda, \mathcal{U}) \rightarrow PP, MSK$ The setup algorithm chooses a bilinear group G of order $N = p_1 p_2 p_3$ (3 distinct primes). We let G_{p_i} denote the subgroup of order p_i in G. It then chooses random exponents $\alpha, a, \kappa \in \mathbb{Z}_N$, a random group element $g_1 \in G_{p_1}$ and a random group element $g_3 \in G_{p_1}$. For each attribute $i \in \mathcal{U}$, it chooses a random value $h_i \in \mathbb{Z}_N$. In addition, we assume that the maximum number of rows of the matrix in any LSSS access structure is bounded by L. The public parameters PP contain $\{N, g = g_1, g^a, g^\kappa, e(g,g)^\alpha, \{H_i = g^{h_i}\}_{i\in\mathcal{U}}, g_3, L\}$. The master secret key MSK additionally contains α.

Outsourcing-KeyGen-PreProc$(PP, MSK) \rightarrow MSK_O, MSK_L$ The outsourcing key generation preprocessing algorithm is run by KGC. It chooses $\alpha_0 \in \mathbb{Z}_N$ randomly, and outputs

$$MSK_O = g^{\alpha_0}, MSK_L = g^{\alpha-\alpha_0}.$$

Outsourcing-KeyGen$(PP, MSK_O, S \subseteq \mathcal{U}) \rightarrow IK_S$ Here S is an attribute set. The outsourcing key generation algorithm is run by OKGSP. It chooses random exponents $t \in \mathbb{Z}_N$, random elements $R, R', R'', \{R_i\}_{i\in S} \in G_{p_3}$, and calculates

$$K_O = g^{\alpha_0} g^{at} R, K'_O = g^t R',$$
$$\{K'_{O,i} = H_i^t R_i\}_{i\in S}.$$

Then, it outputs $IK_S = (S, K_O, K'_O, \{K_{O,i}\}_{i\in S})$.

Outsourcing-KeyGen-Verification$(PP, MSK_O, IK_S) \rightarrow b \in \{0,1\}$ The outsourcing key generation verification algorithm is run by OVSP, when OVSP received a verification request from the KGC. It verifies the following equation:

$$e(K_O, g) \stackrel{?}{=} e(MSK_O, g) e(K'_O, g^a);$$
$$e(K'_O, H_i) \stackrel{?}{=} e(K'_{O,i}, g) \quad \forall i \in S.$$

It outputs $b = 1$ if all of the above equations are satisfied, and $b = 0$ otherwise.

KeyGen$(PP, MSK_L, IK_S) \rightarrow SK_S$ The key generation algorithm is run by KGC. It chooses $u \in \mathbb{Z}_N$ and $\tilde{R}, \tilde{R}', \tilde{R}''$, $\{\tilde{R}_i\}_{i\in S} \in G_{p_3}$ randomly, and calculates

$$K = g^{\alpha-\alpha_0} K_O g^{\kappa\cdot u} \tilde{R}, K' = K'_O \tilde{R}', K'' = g^u \tilde{R}'',$$
$$\{K_i = K_{O,i} \tilde{R}_i\}_{i\in S}.$$

Then, it outputs $SK_S = (S, K, K', K'', \{K_i\}_{i\in S})$.

Outsourcing-Encryption-PreProc$(PP) \rightarrow EP_O, EP_L$ The outsourcing encryption preprocessing algorithm is run by encryptor. For $\forall j \in [1, L]$, it chooses random exponent $\lambda'_j \in \mathbb{Z}_N$, and calculates $ep_j = (g^a)^{\lambda'_j}$.

Then, it outputs $EP_O = \{ep_j\}_{j\in[1,L]}, EP_L = \{\lambda'_j\}_{j\in[1,L]}$.

Outsourcing-Encryption$(PP, EP_O, \mathbb{A} = (\mathbf{M}, \rho)) \rightarrow IC$ Here $(\mathbf{M}, \rho)$ is an access structure, where $\mathbf{M}$ is an $l \times n$ matrix, and ρ is a function, associates rows of $\mathbf{M}$ to attributes. The outsourcing encryption algorithm is run by OESP. For $\forall j \in [1, l]$, it chooses random exponent $r_j \in \mathbb{Z}_N$, and calculates

$$IC_{j,1} = (g^a)^{\lambda'_j} H_{\rho(j)}^{-r_j}, IC_{j,2} = g^{r_j}.$$

Then, it outputs $IC = ((\mathbf{M}, \rho), \{IC_{j,1}, IC_{j,2}\}_{j\in[1,l]})$.

Outsourcing-Encryption-Verification$(PP, EP_O, IC) \rightarrow b' \in \{0,1\}$ The outsourcing encryption verification algorithm is run by OVSP, when OVSP received a verification request from the encryptor. It verifies the following equation:

$$e(IC_{j,1}, g) e(IC_{j,2}, H_{\rho(j)}) \stackrel{?}{=} e(ep_j, g) \quad \forall j \in [1, l].$$

It outputs $b' = 1$ if all of the above equations are satisfied, and $b' = 0$ otherwise.

Encryption$(PP, \text{m}, EP_L, IC) \rightarrow CT$ The encryption algorithm chooses random exponents $x, s, v_2, \ldots, v_n \in \mathbb{Z}_N$, and sets $\boldsymbol{v} = (s, v_2, \ldots, v_n)$, then calculates

$$(\lambda_1, \ldots, \lambda_l) = \mathbf{M} \cdot \boldsymbol{v},$$
$$C_0 = m \cdot e(g,g)^{\alpha\cdot s}, C_1 = g^s, C_2 = (g^\kappa)^s,$$
$$\{C_{j,1} = (g^a)^x IC_{j,1}, C_{j,2} = IC_{j,2},$$
$$C_{j,3} = \lambda_j - \lambda'_j - x\}_{j\in[1,l]}.$$

It outputs

$$CT = ((\mathbf{M}, \rho), C_0, C_1, C_2, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,l]}).$$

Outsourcing-Decryption-PreProc$(SK) \rightarrow (SK_O, SK_L)$ The outsourcing decryption preprocessing algorithm chooses random exponent $\alpha_U \in \mathbb{Z}_N$ and calculates $\bar{K} = K \cdot g^{\alpha_U}$. Then, it outputs

$$SK_L = g^{\alpha_U}, SK_O = (S, \bar{K}, K', K'', \{K_i\}_{i\in S}).$$

Outsourcing-Decryption$(SK_O, CT) \rightarrow CT'$ If S does not satisfy the access structure $(\mathbf{M}, \rho)$, then the algorithm issues an error message. Otherwise, it computes constants $\omega_j \in \mathbb{Z}_N$ such that $\Sigma_{\rho(j)\in S}\omega_j \cdot \mathbf{M}_j = (1, 0, \ldots, 0)$, where $\mathbf{M}_j$ is the jth row of $\mathbf{M}$. Then it calculates and outputs

$$CT' = \frac{e(C_1, \bar{K})e(C_2, K'')^{-1}}{\prod_{\rho(j)\in S}(e(C_{j,1}, K')e(g^{a\cdot C_{j,3}}, K')e(C_{j,2}, K_{\rho(j)}))^{\omega_j}} = e(g, g)^{(\alpha+\alpha_U)\cdot s}$$

Decryption$(CT', SK_L) \rightarrow m$ The decryption algorithm calculates and outputs

$$m = \frac{C_0 \cdot e(C_1, SK_L)}{CT'}.$$

4.2 Comparison

In Table 1, we give the comparison of local computation, where E expresses exponentiation and P expresses pairing. The local computation of our scheme is constant at every stage.

5 Security

In this section, we use the dual-system encryption technique (Waters 2009; Lewko et al. 2010; Lewko and Waters 2012) to prove the adaptive security of our scheme.

At the beginning, we will introduce the definitions of semi-functional keys and semi-functional ciphertexts:

Semi-functional keys Based on the normal key $(K, K', K'', \{K_i\}_{i\in S})$, we choose random elements $W, W'' \in G_{p_2}$, the semi-functional key is defined as:

$$SK^{semi} = (S, KW, K', K''W'', \{K_i\}_{i\in S}).$$

The outsourcing secret key SK_O^{semi} is semi-functional, if it is created using a semi-functional key SK^{semi}.

Semi-functional ciphertexts Based on the normal ciphertext $((\mathbf{M}, \rho), C_0, C, C', \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,l]})$, we randomly choose $a', \kappa', s' \in \mathbb{Z}_N$, $\mathbf{w} \in \mathbb{Z}_N^n$, $\eta'_{\rho(j)}, \gamma'_j \in \mathbb{Z}_N$ for each $j \in [l]$, where s' is the first entry of vector $\mathbf{w}$. The semi-functional ciphertext CT^{semi} is defined as:

$$((\mathbf{M}, \rho), C_0, Cg_2^{s'}, C'g_2^{s'\kappa'}, \{C_{j,1}g_2^{a'\mathbf{M}_j\cdot\mathbf{w}}, C_{j,2}, C_{j,3}\}_{j\in[1,l]}).$$

Then, we will employ the hybrid argument method. The security game $Game_{real}$ is defined in Sect. 3.2. We suppose that the attacker would make Q create key queries. For $k \in [0, Q]$, the kth game $Game_k$ is defined as follow.

$Game_k$ In this game, the intermediate ciphertext and ciphertext are semi-functional, as are the first k keys in $\mathcal{T}_1$, and the other keys are normal.

In this hybrid argument, $Game_{real}$ will be translated to $Game_0$, then to $Game_1$, next to $Game_2$,..., finally to $Game_Q$. Then, $Game_Q$ will be translated to $Game_{final}$, in which the ciphertext is a semi-functional encryption of a random message. That is to say, any attacker has no advantage in this game.

Lemma 1 *Under Assumption 1, no PPT attacker has non-negligible advantage in distinguishing $Game_{real}$ and $Game_0$.*

Proof Suppose there is a PPT attacker $\mathcal{A}$, who has non-negligible advantage in distinguishing $Game_{real}$ and $Game_0$. We can construct a PPT algorithm $\mathcal{B}$ to break Assumption 1. We set $Z_0 := \{1\}$, $Z_1 := \{1, 2\}$, $Z_2 := \{1\}$, $Z_3 := \{3\}$, and give g_1, g_3, T to $\mathcal{B}$, where $g_1 \in G_{p_1}$, $g_3 \in G_{p_3}$, T is either a random element of G_{p_1} or a random element of $G_{p_1p_2}$.

Initial We assume that the maximum number of rows of the matrix in any LSSS access structure is bounded by L. $\mathcal{B}$ chooses $\alpha, \alpha_0, a, \kappa, \{h_i\}_{i\in\mathcal{U}} \in \mathbb{Z}_N$ randomly, and sets $PP = (N, g = g_1, g^a = g_1^a, g^\kappa = g_1^\kappa, e(g, g)^\alpha = e(g_1, g_1)^\alpha, \{H_i = g_1^{h_i}\}_{i\in\mathcal{U}}, g_3, L)$, $MSK_O = g^{\alpha_0}$. PP and MSK_O are given to $\mathcal{A}$. Note, $\mathcal{B}$ has the $MSK = \alpha$ and $MSK_L = g^{\alpha-\alpha_0}$.

Phase 1&2 $\mathcal{B}$ sets two integer counters $\zeta, \xi = 0$, two empty tables $\mathcal{T}_1, \mathcal{T}_2$, and an empty set $\mathcal{D}$. $\mathcal{A}$ can make the following queries to $\mathcal{B}$ repeatedly:

Table 1 Comparison of local computation

Schemes	KeyGen	Encryption	Decryption	Group order	Security
Waters (2011)	$(\|S\|+3)E$	$(3l+2)E$	$(2l+1)P+lE$	Prime	Selective, wRO
Lewko and Waters (2012)	$(\|S\|+5)E$	$(3l+3)E$	$(2l+2)P+lE$	Composite	Adaptive, wRO
Rouselakis and Waters (2015)	$(\|S\|+3)E$	$(6l+1)E$	$3lP+lE$	Prime	Adaptive, RO
Our scheme	$2E$	$4E$	$1P$	Composite	Adaptive, wRO

– *Create (S)* $\mathcal{B}$ can response $\mathcal{A}$'s query on attribute set S by running Outsourcing-KeyGen, KeyGen and Outsourcing-Decryption-PreProc algorithms, because it has MSK, MSK_O and MSK_L.
– *Corrupt (k)* When $\mathcal{A}$ make a corrupt query on index k, $\mathcal{B}$ sets $\mathcal{D} = \mathcal{D} \cup \{S_k\}$, and gives $(SK_{S_k,L}, SK_{S_k})$ to $\mathcal{A}$, if exists a kth entry $(k, S_k, IK_{S_k}, SK_{S_k}, SK_{S_k,O}, SK_{S_k,L})$ in table $\mathcal{T}_1$, and outputs $\perp$ otherwise.
– *Outsourcing-Encryption* ($\mathbb{A}$) $\mathcal{B}$ can response this query by running Outsourcing-Encryption-PreProc and Outsourcing-Encryption algorithm.

Challenge $\mathcal{A}$ submits two messages m_0^*, m_1^* with equal length, and an index j of $\mathcal{T}_2$. If there exists a j^{th} entry $(j, \mathbb{A}^* = (\mathbf{M}^*, \rho^*), EP_O = \{(g^a)^{\lambda'_j}\}_{j\in[1,L]}, EP_L = \{\lambda'_j\}_{j\in[1,L]}, \{(g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j}, g^{r_j}\}_{j\in[1,l]})$ in table $\mathcal{T}_2$, and for all $S \in \mathcal{D}$, $S \notin \mathbb{A}^*$, $\mathcal{B}$ chooses $b \in \{0, 1\}$ randomly, and encrypts m_b as follows. It chooses vector $\tilde{\mathbf{v}} \in \mathbb{Z}_N^n$ randomly, except the first entry is 1, and computes the ciphertext as:

$$C_0 = m_b e(g_1, T)^\alpha, \quad C = T, C' = T^\kappa,$$
$$C_{j,1} = T^{a\mathbf{M}_j^* \cdot \tilde{\mathbf{v}}} (g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j}, C_{j,2} = g^{r_j}, C_{j,3} = -\lambda'_j \ \forall j.$$

Note, $\mathcal{B}$ implicitly sets g^s equal to the G_{p_1} part of T, and $\mathbf{v} = s\tilde{\mathbf{v}}$. If $T \in G_{p_1}$, the ciphertext is normal, and $\mathcal{B}$ simulates $Game_{real}$ with $\mathcal{A}$ properly. If $T \in G_{p_1p_2}$, the ciphertext is semi-functional, $\mathcal{B}$ simulates $Game_0$ with $\mathcal{A}$ properly. Therefore, if $\mathcal{A}$ has non-negligible advantage in distinguishing $Game_{real}$ and $Game_0$, $\mathcal{B}$ can use this advantage to break Assumption 1. □

Lemma 2 *Under Assumption 1, no PPT attacker has non-negligible advantage in distinguishing $Game_{k-1}$ and $Game_k$.*

Proof Suppose there is a PPT attacker $\mathcal{A}$, who has non-negligible advantage in distinguishing $Game_{k-1}$ and $Game_k$ for some $k \in [1, Q]$. We can construct a PPT algorithm $\mathcal{B}$ to break Assumption 1. We set $Z_0 := \{1, 3\}$, $Z_1 := \{1, 2, 3\}$, $Z_2 := \{1\}g$, $Z_3 := \{3\}$, $Z_4 := \{1, 2\}$, $Z_5 := \{2, 3\}$, and give $g_1, g_3, X_1X_2, Y_2Y_3, T$ to $\mathcal{B}$ as input, where $g_1, X_1 \in G_{p_1}$, $X_2, Y_2 \in G_{p_2}$, $g_3, Y_3 \in G_{p_3}$, and T is either a random element of $G_{p_1p_2}$ or a random element of $G_{p_1p_2p_3}$.
Initial We assume that the maximum number of rows of the matrix in any LSSS access structure is bounded by L. $\mathcal{B}$ chooses $\alpha, \alpha_0, a, \kappa, \{h_i\}_{i\in\mathcal{U}} \in \mathbb{Z}_N$ randomly, and sets $PP = (N, g = g_1, g^a = g_1^a, g^\kappa = g_1^\kappa, e(g, g)^\alpha = e(g_1, g_1)^\alpha, \{H_i = g_1^{h_i}\}_{i\in\mathcal{U}}, g_3, L)$, $MSK_O = g^{\alpha_0}$. PP and MSK_O are given to $\mathcal{A}$. Note, $\mathcal{B}$ has the $MSK = \alpha$ and $MSK_L = g^{\alpha-\alpha_0}$.
Phase 1&2 $\mathcal{B}$ sets two integer counters $\zeta, \xi = 0$, two empty tables $\mathcal{T}_1, \mathcal{T}_2$, and an empty set $\mathcal{D}$. $\mathcal{A}$ can make the following queries to $\mathcal{B}$ repeatedly:

– *Create (S)* For the first $k-1$ queries, $\mathcal{B}$ uses the Outsourcing-KeyGen and KeyGen algorithms to generate a normal key $SK_S = \{K, K', K'', \{K_i\}_{i\in S}\}$, and chooses $\tau, \tau' \in \mathbb{Z}_N$ randomly. The semi-functional is formed as:

$$SK_S^{semi} = (K(Y_2Y_3)^\tau, K', K''(Y_2Y_3)^{\tau'}, \{K_i\}_{i\in S})$$

Then, it uses the Outsourcing-Decryption-PreProc algorithm on SK_S^{semi} to obtain corresponding outsourcing secret key.
Note, SK_S^{semi} is distributed properly, as Y_2^τ and $Y_2^{\tau'}$ are distributed as uniformly random elements of G_{p_2}.
– *Corrupt (k)* When $\mathcal{A}$ make a corrupt query on index k, $\mathcal{B}$ sets $\mathcal{D} = \mathcal{D} \cup \{S_k\}$, and gives $(SK_{S_k,L}, SK_{S_k})$ to $\mathcal{A}$, if exists a kth entry $(k, S_k, IK_{S_k}, SK_{S_k}, SK_{S_k,O}, SK_{S_k,L})$ in table $\mathcal{T}_1$, and outputs $\perp$ otherwise.
– *Outsourcing-Encryption* ($\mathbb{A}$) $\mathcal{B}$ can response this query by running Outsourcing-Encryption-PreProc and Outsourcing-Encryption algorithm.

Challenge $\mathcal{A}$ submits two messages m_0^*, m_1^* with equal length, an index j of $\mathcal{T}_2$. If there exists a jth entry $(j, \mathbb{A}^* = (\mathbf{M}^*, \rho^*), EP_O = \{(g^a)^{\lambda'_j}\}_{j\in[1,L]}, EP_L = \{\lambda'_j\}_{j\in[1,L]}, \{(g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j}, g^{r_j}\}_{j\in[1,l]})$ in table $\mathcal{T}_2$, and for all $S \in \mathcal{D}$, $S \notin \mathbb{A}^*$, $\mathcal{B}$ chooses $b \in \{0, 1\}$ randomly, and encrypts m_b as follows. It chooses a random vector $\tilde{v} \in \mathbb{Z}_N^n$, in which the first entry is 1. This means that $g^s = X_1$ and $\mathbf{v} = s\tilde{\mathbf{v}}$. It computes the ciphertext as:

$$C_0 = M_b e(g_1, X_1X_2)^\alpha, \quad C = X_1X_2, \quad C' = (X_1X_2)^\kappa,$$
$$C_{j,1} = (X_1X_2)^{a\mathbf{M}_j^* \cdot \tilde{\mathbf{v}}} (g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j},$$
$$C_{j,2} = g^{r_j}, C_{j,3} = -\lambda'_j \quad j \in [1, l].$$

Note, this means that $g_2^{s'} = X_2$, $\kappa' = \kappa$ modulo p_2, and $a' = a$ modulo p_2. The semi-functional ciphertexts are properly distributed.

For the kth create key query on S, $\mathcal{B}$ chooses $t \in \mathbb{Z}_N$, and $R, R', R'', \{R_i\} \in G_{p_3}$ randomly. It sets:

$$K = (g_1^{\alpha+at} T^\kappa) R,$$
$$K' = g^t R', K'' = T R'', \{K_i = H_i^t R_i\}_{i\in S}.$$

If $T \in G_{p_1p_3}$, K is a normal key. If $T \in G_{p_1p_2p_3}$, K is a semi-functional key. Thus, when $T \in G_{p_1p_3}$, $\mathcal{B}$ simulates $Game_{k-1}$, and when $T \in G_{p_1p_2p_3}$, B simulates $Game_k$. Therefore, if $\mathcal{A}$ has non-negligible advantage in distinguishing $Game_{k-1}$ and $Game_k$, $\mathcal{B}$ can use this advantage to break Assumption 1. □

Lemma 3 *Under Assumption 2, no PPT attacker has non-negligible advantage in distinguishing $Game_Q$ and $Game_{final}$.*

Proof Suppose there is a PPT attacker $\mathcal{A}$, who has non-negligible advantage in distinguishing $Game_Q$ and $Game_{final}$. We can construct a PPT algorithm $\mathcal{B}$ to break Assumption 2. We give $g_1, g_2, g_3, g_1^{\alpha} X_2, g_1^{s} Y_2, T$ to $\mathcal{B}$ as input, where T is either $e(g_1, g_1)^{\alpha s}$ or a random element of G_T.

Initial We assume that the maximum number of rows of the matrix in any LSSS access structure is bounded by L. $\mathcal{B}$ chooses $\alpha, \alpha_0, a, \kappa, \{h_i\} \in \mathbb{Z}_N$ randomly, sets $PP = (N, g = g_1, g^a = g_1^a, g^{\kappa} = g_1^{\kappa}, e(g, g)^{\alpha} = e(g_1, g_1)^{\alpha}, \{H_i = g_1^{h_i}\}_{i \in \mathcal{U}}, L)$, and sets $MSK_O = g^{\alpha_0}$. It gives PP and MSK_O to $\mathcal{A}$. Note, $\mathcal{B}$ knows $MSK = \alpha$ and $MSK_L = g^{\alpha - \alpha_0}$.

Phase 1&2 $\mathcal{B}$ sets two integer counters $\zeta, \xi = 0$, two empty tables $\mathcal{T}_1, \mathcal{T}_2$, and an empty set $\mathcal{D}$. $\mathcal{A}$ can make the following queries to $\mathcal{B}$ repeatedly:

- *Create (S)* $\mathcal{B}$ chooses $t, u, \gamma, \gamma' \in \mathbb{Z}_N$, and $R, R', R'', \{R_i\}_{i \in S} \in G_{p_3}$ randomly, and formed SK_S^{semi} as:

  $$K = (g_1^{\alpha} X_2) g_1^{(at + \kappa u)} R g_2^{\gamma}, K' = g_1^{t} R', K'' = g_1^{u} R'' g_2^{\gamma'}, \{K_i = H_i^{t} R_i\}_{i \in S}$$

  Then, it uses the Outsourcing-Decryption-PreProc algorithm on SK_S^{semi} to obtain corresponding outsourcing secret key.
- *Corrupt (k)* When $\mathcal{A}$ make a corrupt query on index k, $\mathcal{B}$ sets $\mathcal{D} = \mathcal{D} \cup \{S_k\}$, and gives $(SK_{S_k,L}, SK_{S_k})$ to $\mathcal{A}$, if exists a kth entry $(k, S_k, IK_{S_k}, SK_{S_k}, SK_{S_k,O}, SK_{S_k,L})$ in table $\mathcal{T}_1$, and outputs $\perp$ otherwise.
- *Outsourcing-Encryption* $(\mathbb{A})$ $\mathcal{B}$ can response this query by running Outsourcing-Encryption-PreProc and Outsourcing-Encryption algorithm.

Challenge $\mathcal{A}$ submits two messages m_0^*, m_1^* with equal length, an index j of $\mathcal{T}_2$. If there exists a jth entry $(j, \mathbb{A}^* = (\mathbf{M}^*, \rho^*), EP_O = \{(g^a)^{\lambda'_j}\}_{j \in [1,L]}, EP_L = \{\lambda'_j\}_{j \in [1,L]}, \{(g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j}, g^{r_j}\}_{j \in [1,l]})$ in table $\mathcal{T}_2$, and for all $S \in \mathcal{D}$, $S \notin \mathbb{A}^*$, $\mathcal{B}$ chooses $b \in \{0, 1\}$ randomly, and encrypts m_b as follows. It chooses a random vector $\tilde{v} \in \mathbb{Z}_N^n$, in which the first entry is 1. This means that $v = s\tilde{\mathbf{v}}$. It produces the ciphertext as:

$$C_0 = m_b T, \quad C = g_1^{s} Y_2, \quad C' = (g_1^{s} Y_2)^{\kappa},$$
$$C_{j,1} = (g_1^{s} Y_2)^{a \mathbf{M}_j^* \cdot \tilde{\mathbf{v}}} (g^a)^{\lambda'_j} H_{\rho^*(j)}^{-r_j},$$
$$C_{j,2} = g^{r_j}, C_{j,3} = -\lambda'_j \ \forall j.$$

If $T = e(g_1, g_1)^{\alpha s}$, this is a semi-functional ciphertext of m_b, and $\mathcal{B}$ simulates $Game_Q$. If T is a random element of G_T, this is a semi-functional ciphertext of a random message, and $\mathcal{B}$ simulates $Game_{final}$. Therefore, if $\mathcal{A}$ has non-negligible advantage in distinguishing $Game_Q$ and $Game_{final}$, $\mathcal{B}$ can use this advantage to break Assumption 2. □

Theorem 1 *Under Assumptions 1 and 2, the outsourced CP-ABE scheme is secure.*

If Assumptions 1 and 2 hold, Lemmas 1– 3 show that $Game_{real}$ is indistinguishable from $Game_{final}$, in which b is hidden from $\mathcal{A}$. Terefore, $\mathcal{A}$ have no advantage to break $Game_{real}$.

6 Outsourced CP-ABE in cloud computing environment

In this section, we introduce how to deploy our outsourced CP-ABE scheme in cloud computing environment and introduce a specific application for secure cloud storage system.

6.1 Deployment

There are three phases in the running of the outsourced CP-ABE scheme, i.e., key generation phase, encryption phase and decryption phase.

- **Key generation phase** In this phase, KGC issues private keys to users. To calculate the private key effectively, KGC calls the outsourcing key generation service, which is provided by OKGSP-cloud, to share calculation tasks. After receiving the outsourcing results, KGC calls the outsourcing verification service, which is provided by OVSP_{KG}-cloud, to verify the correctness of the calculation. The details are as follows (Fig. 2):

1. KGC runs the **Outsourcing-KeyGen-PreProc** algorithm to obtain (MSK_O, MSK_L) and delegates the MSK_O to OKGSP-cloud, which provide the outsourcing key generation service.

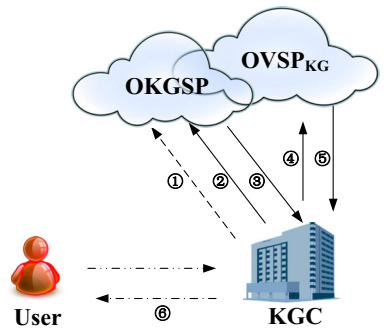

Fig. 2 Key generation phase

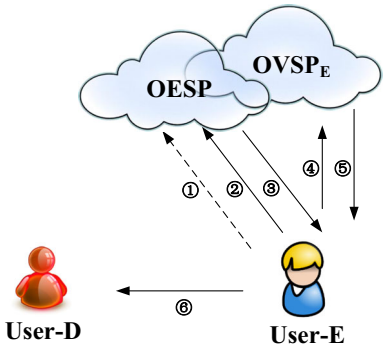

Fig. 3 Encryption phase

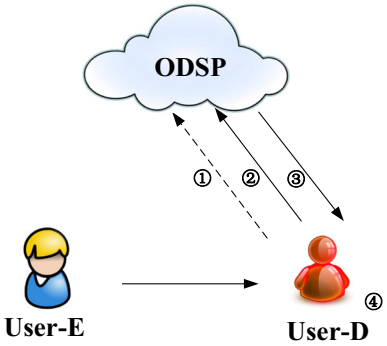

Fig. 4 Decryption phase

2. After KGC receives a key generation request from some user on attributes set S, it submits this request to OKGSP-cloud.
3. According to the request on attribute set S from KGC, OKGSP-cloud runs **Outsourcing-KeyGen** to obtain an intermediate key IK_S, and returns the IK_S to KGC.
4. After receiving IK_S from OKGSP-cloud, KGC submits (MSK_O, IK_S) to OVSP$_{KG}$-cloud to verify the correctness of outsourcing calculation.
5. The OVSP$_{KG}$-cloud runs **Outsourcing-KeyGen-Verification** algorithm on (MSK_O, IK_S) and returns a boolean value to KGC.
6. If IK_S is correct, KGC runs **KeyGen** algorithm on (MSK_L, IK_S) and returns the secret key SK_S to user.

– **Encryption phase** In encryption phase, the encryptor calls the outsourcing encryption service, which is provided by OESP-cloud, to share calculation tasks. After receiving the outsourcing results, encryptor calls the outsourcing verification service, which is provided by OVSP$_E$-cloud, to verify the correctness of the calculation. The details are as follows (Fig. 3):

1. The encryptor runs **Outsourcing-Encryption-PreProc** algorithm to obtain (EP_O, EP_L) and delegates the EP_O to OESP-cloud, which provide the outsourcing encryption service.
2. When encryptor needs to encrypt a message on some access structure $\mathbb{A}$, it submits $\mathbb{A}$ to OESP-cloud.
3. OESP-cloud runs **Outsourcing-Encryption** to obtain an intermediate ciphertext IC and returns the IC to the encryptor.
4. After receiving IC from OESP-cloud, the encryptor submits (EP_O, IC) to OVSP$_E$-cloud to verify the correctness of outsourcing calculation.
5. The OVSP$_E$-cloud runs **Outsourcing-Encryption-Verification** algorithm on (EP_O, IC) and returns a boolean value to the encryptor.
6. If IC is correct, the encryptor runs **Encryption** algorithm on (EP_L, IC) and sends the ciphertexts CT to the decryptor.

– **Decryption phase** In decryption phase, the decryptor calls the outsourcing decryption service, which is provided by ODSP-cloud, to share calculation tasks. After receiving the outsourcing results, decryptor calls the outsourcing verification service, which is provided by OVSP$_D$-cloud, to verify the correctness of the calculation. The details are as follows (Fig. 4):

1. The decryptor runs **Outsourcing-Decryption-PreProc** algorithm to obtain (SK_O, SK_L) and delegates the SK_O to ODSP-cloud, which provide the outsourcing encryption service.
2. When decryptor needs to decrypt a ciphertext CT, it submits CT to ODSP-cloud.
3. OESP-cloud runs **Outsourcing-Decryption** to obtain a partially decrypted ciphertext CT' and returns the CT' to the decryptor.
4. The decryptor runs **Decryption** algorithm on (SK_L, CT') to obtain plaintext m.

6.2 Application in secure cloud storage system

In this section, we constructed a new data access control scheme for cloud storage systems using our outsourced CP-ABE scheme. Figure 5 shows the architecture of secure cloud storage system, which consists of the following system entities:

– **KGC** Its task is to authenticate the user's attributes and to issue corresponding keys for users.
– **Data owner** Data owner can share the data in the cloud storage server for specific users.
– **Cloud storage server** It is a semi-trusted server. It can provide storage space for data owners and allow legitimate users to access data.

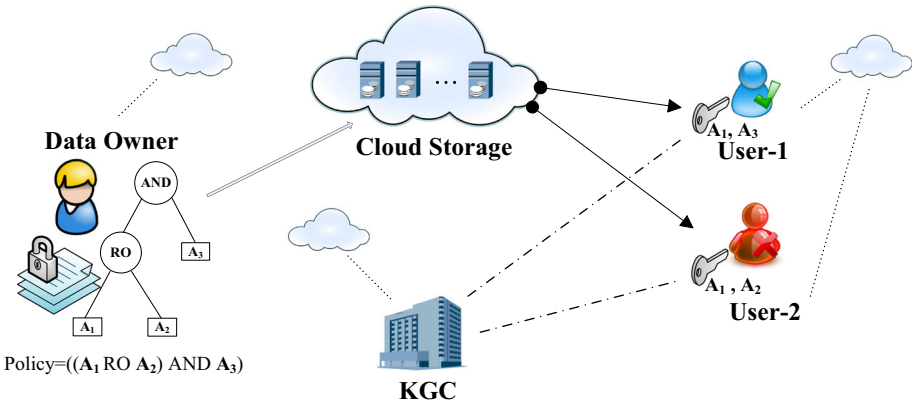

Fig. 5 Secure cloud storage system

– **User** In this system, each user has a set of attribute keys corresponding to their attributes and can access the cloud storage services.

Running of the data access control scheme for cloud storage contains three phases, i.e., key generation phase, encryption phase and decryption phase. They are as described in Sect. 6.1, except that, at the end of encryption phase, data owner stores the ciphertext on the storage cloud.

7 Conclusion

In this paper, we introduced the verifiable outsourced CP-ABE system. In this system, KGC, encryptor and decryptor, are able to outsource their computing tasks to the corresponding service providers, respectively, to improve the computational efficiency, and they are also able to verify the correctness of outsourcing calculation efficiently by using the corresponding outsourcing verification services. Then, we proposed a specific outsourced CP-ABE scheme and prove its adaptive security in the standard model. Finally, we introduced how to deploy our outsourced CP-ABE scheme in cloud computing environment and introduced a specific application for secure cloud storage system.

Acknowledgments This study was funded by the National Natural Science Foundation of China (Grant Numbers 61173139, 61272434, 61502218, 61572294, 61572379), the Natural Science Foundation of Shandong Province (Grant Number ZR2013FQ021), Outstanding Young Scientists Foundation Grant of Shandong Province (Grant Number BS2014DX016), the CICAEET fund, the PAPD fund and the Natural Science Foundation of Hubei Province of China (Grant Number 2015CFB257).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Atallah MJ, Li J (2005) Secure outsourcing of sequence comparisons. Int J Inf Secur 4(4):277–287. doi:10.1007/s10207-005-0070-3

Atallah MJ, Pantazopoulos KN, Rice JR, Spafford EH (2001) Secure outsourcing of scientific computations. Adv Comput 54:215–272. doi:10.1016/S0065-2458(01)80019-X

Beimel A (1996) Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel

Benjamin D, Atallah MJ (2008) Private and cheating-free outsourcing of algebraic computations. In: Sixth annual conference on privacy, security and trust, PST 2008, October 1–3, 2008, Fredericton, New Brunswick, Canada, pp 240–245. doi:10.1109/PST.2008.12

Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: 2007 IEEE symposium on security and privacy (S&P 2007), 20–23 May 2007, Oakland, California, USA, pp 321–334. doi:10.1109/SP.2007.11

Boneh D, Goh E-J, Nissim K (2005) Evaluating 2-dnf formulas on ciphertexts. In: Proceedings of Theory of cryptography, second theory of cryptography conference, TCC 2005, Cambridge, MA, USA, February 10–12, 2005, pp 325–341. doi:10.1007/978-3-540-30576-7_18

Chen X, Li J, Susilo W (2012) Efficient fair conditional payments for outsourcing computations. IEEE Trans Inf Forensics Secur 7(6):1687–1694. doi:10.1109/TIFS.2012.2210880

Chen X, Li J, Ma J, Tang Q, Lou W (2014) New algorithms for secure outsourcing of modular exponentiations. IEEE Trans Parallel Distrib Syst 25(9):2386–2396. doi:10.1109/TPDS.2013.180

Chen X, Huang X, Li J, Ma J, Lou W, Wong DS (2015) New algorithms for secure outsourcing of large-scale systems of linear equations. IEEE Trans Inf Forensics Secur 10(1):69–78. doi:10.1109/TIFS.2014.2363765

Chen X, Susilo W, Li J, Wong DS, Ma J, Tang S, Tang Q (2015) Efficient algorithms for secure outsourcing of bilinear pairings. Theor Comput Sci 562:112—121. doi:10.1016/j.tcs.2014.09.038

Cheung L, Newport C (2007) Provably secure ciphertext policy ABE. In: Proceedings of the 2007 ACM conference on computer and communications security, CCS 2007, Alexandria, Virginia, USA, October 28–31, 2007, pp 456–465. doi:10.1145/1315245.1315302

Fu Z, Kui R, Jiangang S, Xingming S, Fengxiao H (2015) Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEEE Trans Parallel Distrib Syst. doi:10.1109/TPDS.2015.2506573

Fu Z, Sun X, Liu Q, Zhou L, Shu J (2015b) Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEICE Trans 98-B(1):190–200. http://search.ieice.org/bin/summary.php?id=e98-b_1_190

Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM conference on computer and communications security, CCS 2006, Alexandria, VA, USA, October 30–November 3, 2006, pp 89–98. doi:10.1145/1180405.1180418

Green M, Hohenberger S, Waters B (2011) Outsourcing the decryption of ABE ciphertexts. In: Proceedings of 20th USENIX security symposium, San Francisco, CA, USA, August 8–12, 2011. http://static.usenix.org/events/sec11/tech/full_papers/Green.pdf

He D, Zeadally S, Wu L (2015) Certificateless public auditing scheme for cloud-assisted wireless body area networks. IEEE Syst J. doi:10.1109/JSYST.2015.2428620

He D, Kumar N, Shen H, Lee J-H (2016) One-to-many authentication for access control in mobile pay-tv systems. Sci China Inf Sci. doi:10.1007/s11432-015-5469-5

He D, Zeadally S, Kumar N, Lee J-H (2016) Anonymous authentication for wireless body area networks with provable security. IEEE Syst J. doi:10.1109/JSYST.2016.2544805

Huang X, Xiang Y, Bertino E, Zhou J (2014) Robust multi-factor authentication for fragile communications. IEEE Trans Dependable Secure Comput 11(6):568–581. doi:10.1109/TDSC.2013.2297110

Huang X, Liu JK, Tang S, Xiang Y, Liang K, Xu L, Zhou J (2015) Cost-effective authentic and anonymous data sharing with forward security. IEEE Trans Comput 64(4):971–983. doi:10.1109/TC.2014.2315619

Lai J, Deng RH, Guan C, Weng J (2013) Attribute-based encryption with verifiable outsourced decryption. IEEE Trans Inf Forensics Secur 8(8):1343–1354. doi:10.1109/TIFS.2013.2271848

Lewko A, Waters B (2010) New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In: Proceedings of 7th theory of cryptography conference on theory of cryptography, TCC 2010, Zurich, Switzerland, February 9–11, 2010, pp 455–479. doi:10.1007/978-3-642-11799-2_27

Lewko A, Waters B (2012) New proof methods for attribute-based encryption: achieving full security through selective techniques. In: Proceedings of 32nd annual cryptology conference on advances in cryptology-CRYPTO 2012, Santa Barbara, CA, USA, August 19–23, 2012, pp 180–198. doi:10.1007/978-3-642-32009-5_12

Lewko A, Okamoto T, Sahai A, Takashima K, Waters B (2010) Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Proceedings of advances in cryptology-EUROCRYPT 2010, 29th annual international conference on the theory and applications of cryptographic techniques, French Riviera, May 30–June 3, 2010, pp 62–91. doi:10.1007/978-3-642-13190-5_4

Li J, Huang X, Li J, Chen X, Xiang Y (2014) Securely outsourcing attribute-based encryption with checkability. IEEE Trans Parallel Distrib Syst 25(8):2201–2210. doi:10.1109/TPDS.2013.271

Li J, Jia C, Li J, Chen X (2012) Outsourcing encryption of attribute-based encryption with mapreduce. In: Proceedings of 14th international conference on information and communications security, ICICS 2012, Hong Kong, China, October 29–31, 2012, pp 191–201. doi:10.1007/978-3-642-34129-8_17

Ren Y, Shen J, Wang J, Han J, Lee S (2015) Mutual verifiable provable data auditing in public cloud storage. J Internet Technol 16(2):317–324

Rouselakis Y, Waters B (2015) Efficient statically-secure large-universe multi-authority attribute-based encryption. IACR Cryptol ePrint Arch 2015: 16. http://eprint.iacr.org/2015/016

Sahai A, Waters B (2005) Fuzzy identity-based encryption. In: Proceedings of advances in cryptology-EUROCRYPT 2005, 24th annual international conference on the theory and applications of cryptographic techniques, Aarhus, Denmark, May 22–26, 2005, pp 457–473. doi:10.1007/11426639_27

Shen J, Tan H, Moh S, Chung I, Liu Q, Sun X (2015) Enhanced secure sensor association and key management in wireless body area networks. J Commun Netw 17(5):453–462. doi:10.1109/JCN.2015.000083

Shen J, Tan H, Wang J, Wang J, Lee S (2015b) A novel routing protocol providing good transmission reliability in underwater sensor networks. J Internet Technol 16(1):171–178

Wang H, Zheng Z, Lei W, Wang Y (2015) Adaptively secure outsourcing ciphertext-policy attribute-based encryption. J Comput Res Dev 52(10):2270–2280

Waters B (2008) Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. IACR Cryptol ePrint Arch 2008: 290. http://eprint.iacr.org/2008/290

Waters Brent (2009) Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Proceedings of 29th annual international cryptology conference on advances in cryptology-CRYPTO 2009, Santa Barbara, CA, USA, August 16–20, 2009, pp 619–636. doi:10.1007/978-3-642-03356-8_36

Waters B (2011) Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In Proceedings of public key cryptography-PKC 2011-14th international conference on practice and theory in public key cryptography, Taormina, Italy, March 6–9, 2011, pp 53–70. doi:10.1007/978-3-642-19379-8_4