A CP-ABE Access Control Scheme Based on Proxy Re-encryption in Cloud Storage

Haiyong Wang(✉) and Yao Peng

College of Internet of Things,
Nanjing University of Posts and Telecommunications, Nanjing 210003, China
why@njupt.edu.cn

Abstract. With the popular application of cloud storage and the diversification of terminal devices, especially the widespread popularization of smart terminals. Users have more and more requirements for how to access information in the cloud safely and efficiently. Ciphertext policy attribute-based encryption (CP-ABE) is an effective method to achieve fine-grained access control of cloud data. However, the large decryption overhead is a potential problem of attribute-based encryption. In this paper, a CP-ABE access control scheme based on proxy re-encryption is proposed, it helps markedly reduce the user's decryption overhead. Meanwhile, attribute revocation is provided for key update while ensuring fine-grained access control, and an improved decryption key generation method is proposed, which solves the data leakage problem caused by illegal stealing private key in the traditional CP-ABE scheme. A comparison with other CP-ABE schemes shows that our scheme has better decryption performance for mobile devices accessing cloud data.

Keywords: Cloud storage · Attribute-based encryption · CP-ABE
Proxy re-encryption

1 Introduction

With the rapid development of information technology and the explosive growth of user data, it is becoming more and more convenient to share cloud data over the Internet [1]. Cloud computing is one of the most important technologies for cloud data sharing [2], it is a new computing paradigm that provides users with on-demand deployment, dynamic optimization and recovery, on-demand billing, and massive amounts of storage and computing resources available over the Internet at all times and places [3]. Shared data is remote storage, it is a threat to the privacy of the data owner, and the security of the shared data itself [4]. Therefore, enforcing the protection of personal, confidential and sensitive data stored in the cloud is extremely crucial. The simultaneous participation of a large number of users requires fine-grained access control for data sharing [5]. Attribute-based encryption (ABE) [6] provides a flexible access control scheme, it can implement one-to-many communication mechanism in cloud storage environment, which means that a single key can decrypt different ciphertexts or different keys can decrypt the same ciphertext [7]. ABE is divided into two categories: key policy attribute-based encryption (KP-ABE) and ciphertext policy

X. Sun et al. (Eds.): ICCCS 2018, LNCS 11064, pp. 413–425, 2018.
https://doi.org/10.1007/978-3-030-00009-7_38

attribute-based encryption (CP-ABE) [8]. For KP-ABE, the attribute set is associated with the ciphertext, and the access policy is associated with the decryption private key. For CP-ABE, the access policy is embedded into the ciphertext, and the attribute set is embedded into the private key. CP-ABE allows data owner can define their own access policy with an attribute set. Due to this property, CP-ABE is quite suitable for the construction of secure, fine-grained access control for cloud data sharing [1].

In 2005, Sahai and Waters [9] proposed a fuzzy identity-based encryption (FIBE) based on classic identity-based encryption. Since FIBE indicated some many key features of ABE, it laid a theoretical foundation of subsequent research into ABE. In 2006, Goyal *et al.* proposed a formal definition of ABE. With the rapid popularization of mobile intelligent terminals, a growing number of mobile devices participate in cloud data sharing, such as smartphones, wearable devices. Green et al. [10] indicated that ciphertext size and decryption cost are major drawbacks of practical ABE applications. At the same time, there are more and more illegal attacks on mobile terminals at present, and the user's private key is also likely to be stolen by illegal users. A mechanism for deferred re-encryption and proxy re-encryption based on HDFS was proposed in [11]. In this scheme, a large number of encryption and decryption computations are performed by the cloud server and reduces the computational overhead of data owner when the right is revoked. An absolute outsourcing decryption scheme for mobile devices was proposed in [12] to achieve the stability of network traffic of mobile devices. In [13], an attribute-based data sharing system is proposed. By introducing secure two-party computation (2PC) between the key generation center and the data storage center, a new solution to the key escrow problem in a single authorization system is provided.

In this paper, a CP-ABE scheme based on proxy re-encryption (CP-CPE-BPRE) is proposed aiming to solve key escrow problem and decrease decryption overhead. In CP-ABE-BPRE, the private key consists of two parts, the proxy server uses one of the keys to decrypt ciphertext to obtain the semi-decrypted message and sends it to terminal devices, and the devices decrypt the semi-decrypted message by the other key. The scheme can effectively reduce the decryption overhead of user, while guaranteeing flexible and fine-grained access control, but also reduces the data leakage problem caused by the private key exposure.

2 Preliminaries

In this section, we first give some basic definition. Then, we provide formal definitions for access structures and relevant background on Linear Secret Sharing Schemes (LSSS), as taken from [14]. Finally, we will briefly review the cryptographic background about the bilinear map and its security assumption.

Definition 1: The set of users is $U = \{u_1, u_2, \cdots, u_l\}$, where l represents the total number of users.

Definition 2: The set of attributes that have been authorized is $A = \{att_1, att_2, \ldots\ldots att_p\}$, where p represents the total number of attributes.

Definition 3: Let G_t denotes users with the same attribute att_t, and $G_t \subseteq U$. The collection of attribute group is represented as $\mathcal{G} = \{G_t \mid \forall att_t \in A\}$.

Definition 4: Let K_t denote the attribute group key associated with G_t. The collection of attribute group key is represented as $\mathcal{K} = \{K_t \mid \forall att_t \in A\}$.

Definition 5 (Access Structure): Let $P = \{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^P$ is monotone if for arbitrary B and C we have that $B \in \mathbb{A}$ and $B \subseteq C$, and $C \in \mathbb{A}$ holds. An access structure (monotone access structure) is a collection (monotone collection) $\mathbb{A} \subseteq 2^P | \{\Phi\}$. We call sets in $\mathbb{A}$ authorized sets, and sets not in $\mathbb{A}$ unauthorized sets.

Definition 6 (Linear Secret Sharing Schemes, LSSS): Let P be a set of participants and M be a matrix with m rows and d columns. The map $\rho : \{1, 2, \cdots, m\} \to P$ associates each row with one participant for labeling. A secret sharing scheme Π over P for access structure $\mathbb{A}$ is a linear secret sharing scheme in Z_p^* represented by (M, ρ) if it consists of two polynomial-time algorithms:

Share (M, ρ): The share algorithm takes $s \in Z_p^*$ as an input secret to be shared. Share randomly chooses a group of elements $r_1, r_2, \cdots, r_n \in Z_p^*$ and generates a column vector $\vec{v} = (s, r_1, r_2, \cdots, r_n)^T$. Then, it outputs $M \cdot \overrightarrow{v}$ as a vector that l participants share such that they each possess an element. We define M_i as the ith row in M so that $\lambda_{\rho(i)} = M_i \cdot \overrightarrow{v}$ is the element belonging to participant $\rho(i)$.

Recovery (S): The recovery algorithm takes a set S of participants as input. We define a set $I = \{i : \rho(i) \in S\} \subset \{1, 2, \cdots, m\}$. If $S \in \mathbb{A}$, there exists a group of constants $\left\{\omega_i \in Z_p^*\right\}_{i \in I}$, it can recovery the shared secret by $\sum \omega_i \cdot \rho_{(i)} = s(i \in I)$.

Definition 7 (Bilinear pairings): Let $\mathbb{G}_0$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ be a bilinear map with the properties:

- Bilinearity: For all $u, v \in \mathbb{G}_0$ and $a, b \in Z_p$, we have $e\left(u^a, v^b\right) = e(u, v)^{ab}$.
- Non-degeneracy: There exists a generator g of $\mathbb{G}_0$ such that $e(g, g) \neq 1$ holds.
- Computability: For all $u, v \in \mathbb{G}_0$, there exists an effective calculation $e(u, v)$.

Assumption 1 (Discrete Logarithm Assumption): Given two $X, Y \in \mathbb{G}_0$, no probabilistic polynomial-time algorithm can find an integer $k \in Z_p^*$ with a non-negligible advantage for which $Y = X^k$ holds.

Assumption 2 (Decisional Bilinear Diffie-Hellman Assumption): For an arbitrary group of exponents $a, b, c \in Z_p^*$, given a tuple $\left(g, g^a, g^b, g^c, \mathrm{e(g,g)}^{abc}\right)$, no probabilistic polynomial-time algorithm can find an integer $z \in Z_p^*$ with a non-negligible advantage for which $e(g, g)^z = e(g, g)^{abc}$ holds.

3 CP-ABE Access Control Scheme Based on Proxy Re-encryption

The above introduced the existing scheme deficiencies and the basic concepts. This section details how to improve the scheme and reduce the decryption overhead.

3.1 Model Construction

The scheme proposed in this paper is shown in Fig. 1, which is mainly composed of 5 parts. The 5 components communicate through the Internet.

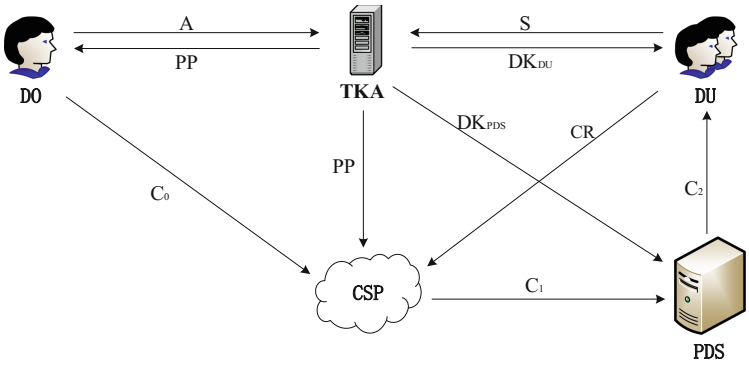

Fig. 1. CP-CPE-BPRE scheme model

- Trusted Key Authority (TKA): The TKA is a vital component in the system. The KA is responsible for most computing tasks, including key generation, key update, etc. The TKA is considered credible in our system.
- Data Owner (DO): DO is an authorized user who possesses data to be uploaded. DOs define their own explicit access policies. Only users whose attributes meet the access policy can obtain plaintext.
- Cloud Service Provider (CSP): CSP is responsible for data storage, management and ciphertext re-encryption. We assume that the CSP is semi-trusted in our system, meaning it is curious about the value of plaintext but has no intention of tampering with it.
- Proxy Decryption Server (PDS): PDS undertakes most of the decryption calculations in our model, it is also considered to be semi-trusted.
- Data User (DU): DO is an authorized user who accesses ciphertext. With the rapid development of cloud services, mobile devices have become the main devices for accessing cloud data. Therefore, we think there is a possibility that the private key of the terminal may be stolen.

We assume that all entities involved in data sharing does not collude with each other to access data illegally, otherwise the system model does not make any sense. The attributes submitted by users are authenticated by TKA. The user's attribute set is a set

that uniquely identifies the user. Each authorized attribute is represented by a random number in public parameters (PP). When DO wants to upload data, it encrypts the data with its access policy and public parameters to obtain C_0. Then, C_0 is uploaded to the cloud server, which is re-encrypted to C_1 by the cloud server. When DU submits an attribute to TKA, TKA generates a private key composed of two parts, which are respectively distributed to DU and PDS. After the DU sends a ciphertext request (CR) to the CPS, the CPS sends C_1 to PDS. PDS decrypts C_1 using the key from TKA to obtain C_2, which can be decrypted by DU.

3.2 Scheme Construction

1. Setup

$\mathbb{G}_1$ and $\mathbb{G}_2$ are two multiplicative cyclic groups with prime order p. Let k be a security parameter and g be a generator of $\mathbb{G}_1$. The hash functions are defined as follows:

$$H : \{0,1\}^* \rightarrow \mathbb{G}_1$$

$$H_1 : \mathbb{G}_2 \rightarrow Z_P^*$$

TKA completes the initialization operation and returns a group of public parameters and master key of system.

Algorithm 1: $Setup\left(1^k, A\right)$

Input: security parameter k and authorized attribute set A

Output: PP (public parameters) and MK (master key)

Step1: TKA selects a group of random elements $h_1, h_2, \cdots, h_m \in_R \mathbb{G}_1$ and $\alpha, \beta \in Z_p^*$, where the element h_t corresponds with an authorized attribute att_t $\left(t \in [1, m]\right)$.

Step2: outputs the following public parameters and master key:

$$PP = \left\{g, g^{\beta}, h_1, h_2, \ldots\ldots, h_m, e(g,g)^{\alpha}, H, H_1\right\}$$

$$MK = \left\{\beta, g^{\alpha}\right\}$$

2. Key Generation

This algorithm is run by the TKA, takes as input PP, MK and the attribute collection submitted by the DU. The private key is composed of DK_{PDS} and DK_{DU}, which are respectively saved by the PDS and the DU, the PDS cannot decrypt ciphertext alone.

Algorithm 2: $KeyGen(PP, MK, S)$

Input: PP, MK and S (the attribute collection submitted by the DU)

Output: DK_{PDS} and DK_{DU}

Step1: chooses a random exponent $\tau \in_R Z_p^*$. Then, computing the initial key $K_3 = g^{(\alpha+\beta\tau)r/\pi}$.

Step2: computes $K_2 = g^{\alpha+\beta\tau}$ using the random number α, β that generated the public parameters and τ in Step1.

Step3: chooses random exponents $r, \pi, \in_R Z_p^*$. Then, computing $K_3 = g^{(\alpha+\beta\tau)r/\pi}$.

Step4: uses the random number in Step3 to calculate $D = g^{\tau r/\pi}, \forall x \in S : D_x = h_x^{\tau r/\pi}$.

Step5: outputs private key as follows:

$$DK_{DU} = \pi / r$$

$$DK_{PDS} = \left\{ K_3 = g^{(\alpha+\beta\tau)r/\pi}, D = g^{\tau r/\pi}, \forall x \in S : D_x = h_x^{\tau r/\pi} \right\}$$

3. Encryption

The encryption algorithm, which is run by DO, takes as input the public parameter *PP*, an access structure $\mathbb{A}$ and plaintext $\mathcal{M}$ and guarantees the confidentiality and integrity of the data.

Algorithm 3: $Encrypt(PP, \mathcal{M}, \mathbb{A})$

Input: PP, $\mathbb{A}$ and $\mathcal{M}$

Output: initial ciphertext C_0

Step1: generates vector $\vec{v} = (s, y_2, y_3, \ldots\ldots, y_d)$ and linear secret sharing scheme (M, ρ).

Step2: computes $\lambda_i = \vec{v} \cdot M_i$ for each user $u_i \in U$.

Step3: the ciphertext is published as:

$$C_0 = \left((M, \rho), C = \mathcal{M} \cdot e(g, g)^{\alpha s}, C' = g^s, \ \forall att_t \in \mathbb{A}: \ C_t = g^{\beta\lambda_t} h_{\rho(t)}^{-s} \right)$$

4. Re-encryption

The re-encryption algorithm, which is run by CSP, takes as input the public parameter *PP*, initial ciphertext C_0, and the collection of attribute groups $\mathcal{G}$. When CSP receives C_0 and $\mathcal{G}$, the re-encryption algorithm selects two random numbers $\mu, \gamma \in_R Z_p^*$ to construct attribute group key set $\mathcal{K}$, and defines a unique identifier $ID_k \in \{0,1\}^*$ for user $u_k \in \mathcal{G}$, which will not change with user attributes. We adopt the attribute group-based algorithm of [13] to re-encrypt the initial ciphertext.

Algorithm 4: $\mathrm{Re}\,Encrypt\left(PP, C_0, \mathcal{G}\right)$

Input: PP, C_0 and $\mathcal{G}$

Output: ultimate ciphertext C_1

Step1: defines a unique identifier $ID_k \in \{0,1\}^*$ for user $u_k \in \mathcal{G}$.

Step2: selects two random numbers $\mu, \gamma \in_R Z_p^*$. Then, computing the session key of $SK_{DU_k} = H\left(ID_k\right)^{\gamma}$ and $SK_{CSP} = g^{\gamma}$ respectively for the DU and the CSP and the corresponding element $x_k = H_1\left(e\left(Q_k^{\mu}, SK\right)\right)$ for each DU.

Step3: constructs a polynomial $f_t(x) = \prod_{i=1}^{v}(x - x_i) = \sum_{i=0}^{v} a_i x^i \pmod p$ for each $G_t \in \mathcal{G}$, where v represents the number of DU in G_t.

Step4: defines $\{P_0, P_1, \cdots, P_v\} = \left\{g^{a_0}, g^{a_1}, \cdots, g^{a_v}\right\}$, and choosing random number $R \in_R Z_p^*$ to computing $Head_t = \left(K_t \cdot P_0^R, P_1^R, \cdots, P_v^R\right)$.

Step5: builds a header message $Head = \left(g^{\mu}, \gamma, \forall G_t \in \mathcal{G} : Head_t\right)$.

Step6: the ultimate ciphertext is published as:

$$C_0' = \left((M, \rho), C = \mathcal{M} \cdot e(g,g)^{\alpha s}, C' = g^s, \forall att_t \in \mathbb{A}: \; C_t = \left(g^{\beta \lambda_t} h_{\rho(t)}^{-s}\right)^{K_t} \right)$$

$$C_1 = \left(Head, C_0'\right)$$

5. Decryption

The decryption algorithm takes as input DU's attribute set S, the ultimate ciphertext C_1, and all private key components DK_{PDS} and DK_{DU}. On receiving a ciphertext request from a DU u_k, the CSP and TKA send the ultimate ciphertext C_1 and the private key DK_{PDS} to the PDS immediately. The PDS decrypts C_1 outputs the partially decrypted ciphertext C_2. The PDS sends C_2 to u_k, and u_k decrypts it with its own private key DK_{DU}.

Algorithm 5: $Decrypt\left(S, C_1, DK_{PDS}, DK_{DU}\right)$

Input: S, C_1, DK_{PDS} and DK_{DU}

Output: plaintext $\mathcal{M}$

Step1: the PDS constructs the attribute group key $\left\{K_t \mid \forall \theta_t \in S\right\}$ as follows:

$$H_1\left(e\left(SK_{DU_k}{}^{\mu}, SK_{CSP}\right)\right) = x_k$$

$$K_t \cdot P_0^R \prod_{i=1}^{v} P_i^{R \cdot x_k^i} = K_t$$

Step2: the PDS extracts an elements C_2 as follows:

$$\begin{aligned}
&\prod_{\rho(t)\in S}\left(e\left(C_t^{1/K_t}, D\right)\cdot e\left(C', D_{\rho(t)}\right)\right)^{\omega_t} \\
&= \prod_{\rho(t)\in S}\left(e\left(\left(g^{\beta\lambda_t} h_{\rho(t)}^{-s}\right)^{K_t\cdot 1/K_t}, g^{\tau r/\pi}\right)\cdot e\left(g^s, h_{\rho(t)}^{\tau\, r/\pi}\right)\right)^{\omega_t} \\
&= \prod_{\rho(t)\in S}\left(e(g,g)^{\beta\lambda_t \tau r/\pi}\right)^{\omega_t} \\
&= e(g,g)^{\beta s\tau r/\pi} \\
&= C_1'
\end{aligned}$$

$$\begin{aligned}
&e\left(C', DK_{PDS}\right)/C_1' \\
&= e\left(g^s, g^{(\alpha+\beta\tau)r/\pi}\right)/e(g,g)^{\beta s\tau r/\pi} \\
&= e(g,g)^{\alpha sr/\pi} \\
&= C_2
\end{aligned}$$

Step3: on receiving C_2 from the PDS, the u_k extracts plaintext as follows:

$$\begin{aligned}
&C/C_2{}^{DK_{DU}} \\
&= \mathcal{M}\cdot e(g,g)^{\alpha s}/\left(e(g,g)^{\alpha sr/\pi}\right)^{\pi/r} \\
&= \mathcal{M}
\end{aligned}$$

4 Comprehensive Analysis

This section mainly analyses the proposed scheme CP-ABE-BPRE in this paper from attribute revocation, security requirements and efficiency. In our analysis, we use notation shown in Table 1.

4.1 Revocation

When a user comes to hold or drop an attribute, the private key should be updated to prevent the user from accessing the previous or subsequent encrypted data for backward or forward secrecy. On receiving a join or leave request for some attribute groups

from a user (e.g., a user comes to hold or drop an attribute att_q at some time instance), the proposed CP-ABE-BPRE will execute the revoking operation immediately. At the beginning of revocation, the KA updates the attribute group from $\mathcal{G}$ to $\mathcal{G}'$ and the attribute group key of G_q and sends the updated membership list of the attribute group to CSP. The CSP runs the re-encryption algorithm upon receiving C_0 and $\mathcal{G}'$. The updated attribute group key set is:

$$\mathcal{K}' = \left\{ K_q', \forall att_t \in A \backslash \left\{ att_q \right\} : K_t \right\}$$

The CSP constructs a new polynomial for G_q, where v' represents the number of DU in G_q:

$$f_q(x) = \prod_{i=1}^{v'} (x - x_i) = \sum_{i=0}^{v'} a_i' x^i (\mathrm{mod}\, p)$$

Then, the re-encryption algorithm selects a new random number $R' \in_R Z_p^*$ to build a new header message as follow:

$$\{P_0, P_1, \cdots, P_{v'}\} = \left\{ g^{a_0'}, g^{a_1'}, \cdots, g^{a_v'} \right\}$$

$$Head_q' = \left(K_q' \cdot P_0^{R'}, P_1^{R'}, \cdots, P_{v'}^{R'} \right)$$

$$Head' = \left(g^{\mu}, Head_q', \forall G_t \in \mathcal{G} \backslash \left\{ G_q \right\} : Head_t \right)$$

Subsequently, the CSP re-encrypts the ciphertext as:

$$C_0'' = \left\{ C = \mathcal{M} \cdot e(g,g)^{\alpha s}, C' = g^s, C_q = \left(g^{\beta \lambda_q} h_{\rho(q)}^{-s} \right)^{K_q} \right.$$
$$\left. (M, \rho),\ \forall att_t \in \mathbb{A} \backslash \left\{ att_q \right\} :\ C_t = \left(g^{\beta \lambda_t} h_{\rho(t)}^{-s} \right)^{K_t} \right\}$$

$$C_1'' = \left(Head', C_0'' \right)$$

Finally, the key generation algorithm is implemented to update the privacy key:

$$DK_{DU}' = \pi' \Big/ r'$$

$$DK_{CSP}' = \left\{ K_3' = g^{(\alpha + \beta\tau) r'/\pi'}, D = g^{\tau r'/\pi'}, \forall x \in S : D_x = h_x^{\tau r'/\pi'} \right\}$$

Table 1. Notations relevant to efficiency comparision.

Symbol	Definition
$\|A\|$	Sum of attributes in an access structure
$\|S\|$	Sum of attributes held by a DU
S_T	Sum of threshold value of all "AND" gates in an access structure
S_G	Sum of gates in an access tree
v	Sum of DUs in a group attribute
t_h	A calculation of hash function
t_p	A calculation of pairing
t_e	A calculation of exponentiation
t_m	A calculation of multiplication
t_d	A calculation of division

4.2 Security Requirements

For the CP-ABE-BPRE scheme, we assume that TKA is trusted, the CSP and PDS are honest but curious about plaintext, and parties do not collude to access data illegally. In this scheme, the CSP re-encrypts the initial ciphertext. Like the trusted third party in [8, 10], the TKA generates and distributes the private key by the submitted attributes of user. However, the difference lies in that the private key of the user in the CP-ABE-BPRE scheme is composed of two parts, respectively stored in the PDS and the DU, and the PDS cannot decrypt the ciphertext only by the key kept in itself. In the process of PDS decrypting C_1 to C_2, since C_2 is encrypted by ElGamal algorithm, the PDS cannot obtain the plaintext. $C_1^{'}$ and C_2 cannot be accessed for user whose attributes do not satisfy the access policy.

In terms of backward and forward secrecy, we adopt an attribute group mechanism inspired [13] to guarantee backward and forward secrecy. Once a DU lose an attribute, the TKA will update the attribute group list immediately. Then, the CSP updates the set of attribute group keys for re-encrypting ciphertext by rebuilding the head message at once. Later, the TKA distributes updated private key components. For ciphertext accessible only to those who hold this attribute, those who no longer possess this attribute can by no means extract any information from the ciphertext. If a DU obtains a new attribute, the CSP updates the attribute group key. Then, it embeds the newly rebuilt head message into the ciphertext. Finally, all DUs in the new attribute group obtain new private key components. Even if this DU possesses ciphertext encrypted before acquiring this attribute, it cannot be correctly decrypted by this DU. Thus, backward and forward secrecy can be guaranteed.

For key exposure, the private key is directly stored in the mobile terminal [8]. When the private key is obtained by the unauthorized user, the illegal user can directly decrypt the ciphertext. In [10], the proxy decryption server is also introduced to reduce the user's decryption cost. However, the private key of the proxy decryption server is generated and distributed by the client. Therefore, the risk of private key exposure remains. The problem that the terminal private key may be stolen is solved by introducing secure two-party computation in [13], but there is a problem that the terminal

has a larger decryption overhead. Mobile devices, such as smartphones, are far inferior to cloud storage servers in privacy protection, which may lead to the exposure of users' private keys stored in these devices. In the CP-ABE-BPRE scheme, the private key is composed of two parts, which are respectively stored at the PDS and DU. When the terminal's key is stolen illegally, unauthorized users can not directly decrypt the ciphertext stored in the PDS, thus effectively preventing the user from disclosing the key exposure problem.

4.3 Efficiency

In terms of efficiency, we assume that the time delay in this scheme is acceptable to the user and compare the CP-ABE-BPRE scheme with the [8, 10, 13]. Each scheme is compared in terms of total decryption overhead and client decryption overhead, the comparison results are presented in Tables 2 and 3. In CP-ABE, the calculation of bilinear pairs takes up most of the computational overhead. The introduction of attribute group encryption mechanism in CP-ABE-BPRE scheme adds an amount of computational overhead. However, most of the decryption computations are performed by PDS through proxy decryption, which obviously reduces user's decryption overhead. As shown in Table 2, compared with other schemes, the total decryption overhead in the CP-ABE-BPRE does not have obvious advantages. However, when the user decryption overhead is significantly reduced, server-side decryption overhead is appropriately increased, which is acceptable. As can be seen from Table 3, the CP-ABE-BPRE is similar to the scheme in [10], the mobile terminal only needs to perform an exponentiation and division operation to complete the decryption. However, the private key generation and distribution of the proxy decryption server in [10] adds the extra computation overhead of client. In the schemes [8, 13], the user has to bear all the computational decryption overhead in the system, which is not conducive to the safe and efficient access of the mobile terminal to the cloud storage data.

Table 2. Comparison of total decryption overhead

Scheme	Decryption overhead
[8]	$(2\|A\|+1)t_p + S_T \cdot t_e + (S_T - S_G)t_m + 2t_d$
[10]	$(\|S\|+2)t_p + (2\|S\|+1)t_e + (2\|S\|-1)t_m + 2t_d$
[13]	$1 \cdot t_h + (2\|S\|+2)t_p + (S_T+v)t_e + (S_T - S_G + v)t_m + 2t_d$
CP-ABE-BPRE	$1 \cdot t_h + (2\|S\|+2)t_p + (2\|S\|+v+1)t_e + (2\|S\|+v-1)t_m + 2t_d$

Table 3. Comparison of client decryption overhead

Scheme	Decryption overhead
[8]	$(2\|A\|+1)t_p + S_T \cdot t_e + (S_T - S_G)t_m + 2t_d$
[10]	$1 \cdot t_e + 1 \cdot t_d$
[13]	$1 \cdot t_h + (2\|S\|+2)t_p + (S_T+v)t_e + (S_T - S_G + v)t_m + 2t_d$
CP-ABE-BPRE	$1 \cdot t_e + 1 \cdot t_d$

In summary, our scheme has better decryption performance for mobile terminals accessing cloud data compared with the schemes of [8, 10, 13].

5 Conclusion and Future Work

Ciphertext policy attribute-based encryption achieves fine-grained access control in cloud storage. A CP-ABE scheme based on proxy re-encryption is proposed in this paper, which introduces the concept of attribute groups in [13] to implement ciphertext re-encryption. The proposed scheme perfectly optimizes clients' user experience since only a small amount of responsibility is taken by them for decryption. Meanwhile it addresses a worse problem called key exposure. Thus, the proposed scheme performs better in cloud data sharing system serving massive performance-restrained mobile devices with respect to either security or efficiency.

The user's encryption overhead and the size of the ciphertext are also the areas of great attention in the attribute encryption technology. We expect to improve the scheme by reducing the encryption overhead, total decryption overhead and ciphertext size of the system in our future work.

Acknowledgments. This research is supported by Education Information Research funded topic in Jiangsu Province (20172105), Nanjing University of Posts and Telecommunications Teaching Reform Project (JG06717JX66) and the special topic of Modern Educational Technology Research in Jiangsu province (2017-R-59518). The authors thank the sponsors for their support and the reviewers for helpful comments.

References

1. Sukhodolskiy, I.A., Zapechnikov, S.V.: An access control model for cloud storage using attribute-based encryption. In: Young Researchers in Electrical and Electronic Engineering, pp. 578–581. IEEE (2017)
2. Wang, S., Zhou, J., Liu, J.K., et al.: An efficient file hierarchy attribute-based encryption scheme in cloud computing. IEEE Trans. Inf. Forensics Secur. **11**(6), 1265–1277 (2016)
3. De, S.J., Ruj, S.: Efficient decentralized attribute based access control for mobile clouds. IEEE Trans. Cloud Comput. **PP**(99), 1 (2017)
4. Sun, G., Dong, Y., Li, Y.: CP-ABE based data access control for cloud storage. J. Commun. **32**(7), 146–152 (2011)
5. Yang, G., Wang, D.-Y., Zhang, T., et al.: Attribute-based access control with multi-authority structure in cloud computing. J. Nanjing Univ. Posts Telecommun. (Nat. Sci.) **34**(2), 1–9 (2014)
6. Goyal, V., Pandey, O., Sahai, A., et al.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98. ACM (2006)
7. Yan, X., Meng, H.: Ciphertext policy attribute-based encryption scheme supporting direct revocation. J. Commun. **37**(5), 44–50 (2016)
8. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334. IEEE Computer Society (2007)

9. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
10. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of ABE ciphertexts. In: Usenix Conference on Security, p. 34. USENIX Association (2011)
11. Zhang, R., Chen, P.S.: A dynamic cryptographic access control scheme in cloud storage services. J. Inf. Process. Manag. **4**(1), 50–55 (2012)
12. Ohigashi, T., Nishimura, K., Aibara, R., et al.: Implementation and evaluation of secure outsourcing scheme for secret sharing scheme on cloud storage services. In: Computer Software and Applications Conference Workshops, pp. 78–83. IEEE (2014)
13. Hur, J.: Improving security and efficiency in attribute-based data sharing. IEEE Trans. Knowl. Data Eng. **25**(10), 2271–2282 (2013)
14. Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_4