More Secure Outsourcing CP-ABE Scheme Under the Situation of Untrusted PKG

Fang Zheng[1,*], Pingzhen Li[1], Xinguang Peng[2], Zhidan Li[3]

[1]Department of Information, Shanxi University of Finance and Economics, Taiyuan 030006, China
[2]Department of Computer and Science, Taiyuan University of Technology, Taiyuan 030024, China
[3]The 36th Research Institute of China Electronics Technology Corporation of Zhejiang Province, Hangzhou 314414, China

Abstract: The scheme of CP-ABE is widely used in data security protection in cloud outsourcing service. The architecture based on CP-ABE involves user, SP, PKG and CSP. Under this situation, PKG is trusted by default. However, in the real situation, except the default situation, there exists another scenario, where there is no SP and user communicates with PKG and CSP directly. In the above environment PKG may be untrusted. Once PKG is colluded with CSP or PKG is compromised, the adversary can get the decryption keys from PKG easily and the data preserved on CSP will be decrypted and leaked. In this paper, we first innovatively propose a scheme that can resist the attack from the aforementioned situations. The idea of the scheme is that a factor generated by the user is added to the secret key and the cipher text. The secret key is added the factor, which results that even PKG is compromised or untrusted, the secret key cannot be obtained. We apply the idea on the Li's scheme and make it be suitable for the cloud outsourcing environment with an untrusted PKG. Finally, we prove the security of our scheme.

Keywords: Outsourcing; CP-ABE; PKG; Safe Factor; CSP

DOI: 10.57237/j.cst.2023.04.007

1 Introduction

The introduction of ABE solves the privacy problem in cloud outsourcing services which makes the users safely upload their data into CSP (Cloud Server Provider). The architecture of the service model is shown in Figure 1. In these scenarios, SP (Service Provider) may be the server of a campus network or the server of a government website or the server of banking. They want to store their data to CSPs and set up PKG (Primary Key Generation) server by themselves, i.e. SP and PKG belong to the same service provider, in which PKG is absolutely trusted by SP and it's no possible that PKG leaks the secret to CSPs. But there exists another application scenario shown in Figure 2 in which the user interacts with CSP directly and in this situation, PKG is a special and independent service provider to serve for encryption and secret key management without being controled by any other SPs or users, where PKG is not trusted completely and can be shared by a lot of users (including SPs). In this scenarios, whoever the users or SPs, they are all be viewed as the clients and PKG is a standalone server to provide the corresponding cryptographical work for the clients. To be specific, the function of PKG is to generate the initiative parameters and the private key of decryption for the cryptosystem. For example, the user wants to copy his/her telephone directory to CSP or uploads his/her diaries or some data related to his/her work to CSP and wants to

Funding: Shanxi soft science research project (grant number: 2018041039-2); Teaching reform project of Shanxi University of Finance and Economics (grant number: 2018226); Research project of Shanxi statistical society (grant number: KY (2019) 164).

*Corresponding author: Fang Zheng, alice20110324@126.com

Received: October 22, 2023; Accepted: December 6, 2023; Published Online: December 9, 2023 http://www.computscitech.com

encrypt the data by the algorithm of ABE. The Users or SPs encrypt their data and send them to PKG and then ask PKG to generate the private keys for the decryption. But PKG is not completely trusted. If PKG and CSP collude, i.e. PKG provides the private keys to CSP, CSP will have the ability to use the algorithm of fuzzy identity-based encryption [1] or CP-ABE [2-4, 6] etc. to decrypt the user's cyphertext. The user's privacy will be breached.

To this scenario, we propose a scheme to solve it by adding a factor which is known only by the user itself, into the cyphertext and the private keys. When the user encrypts the data, he/she will add a factor into C_i which is one part of the cyphertext. But this operation will result that the cyphertext is changed and cannot be decrypted by the original private keys. So, when the user receives the private keys from PKG to prepare for decryption, he/she will add the factor into every private key, which can make the ciphertext be decrypted. The work flow of the idea is shown in Figure 3 and Figure 4. The user keeps the secret factor and both PKG and CSP don't know it. Therefore, even PKG and CSP collude, the cyphertext cannot be decrypted yet.

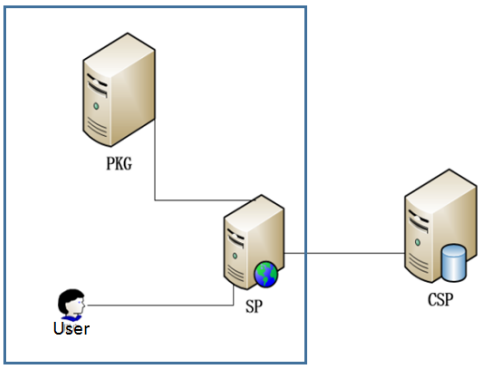

Figure 1 The architecture of PKG affiliate to SP

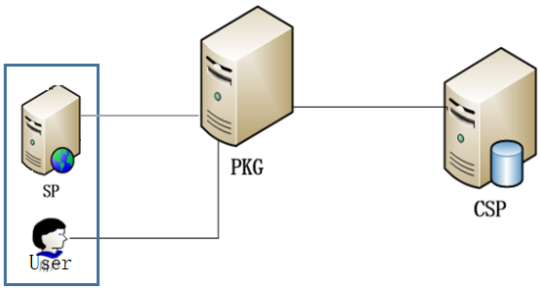

Figure 2 The architecture of PKG is a standalone server

The contributions of our paper are as follows:

1. We propose a scheme of adding a secret factor into the cyphertext and the private keys to prevent PKG and CSP collude;
2. Our scheme provides a secure method for the personal user to upload their data on the cloud server irrespective of whether PKG is secure or not;
3. We apply this idea to improve Li's scheme [25] to make it suitable for the second scenarios;
4. We prove our scheme secure.

The organization of the rest paper is as follows:

The second part is related works; the third is preliminaries; the fourth is the overview of our scheme; the fifth is the detailed description of our scheme; the sixth proves the security of our scheme; the seventh analyzes the performence. The eighth is conclusion.

2 Related Work

In 2005 A Sahai and B Waters [1] propose the fuzzy identity-based encryption which embody the idea of ABE [20]. KP-ABE [5, 14-16] and CP-ABE [6, 17-19] are put forward in sequence. Frikken K et al. [9] propose ABE based on hidden policy to protect the user's identity, which makes the CSPs don't know who visit the ciphertext for the attribute sets the users owe belong to the same attribute category sets. But the encryption and decryption efficiency of the algorithm is a bottleneck, especially decryption occupying a lot of time and add huge burdens for the client device. So how to reduce the computing burden on the client device becomes the research spot of the subject. In 2011 Matthew Green et al. [7] propose that it is the CSPs that shares the partial computing tasks in the process of decryption by introducing a transformation key to reduce the overhead of the mobile device. Chim T W et al. [10] propose a scheme of outsourcing the encryption process to the cloud using the technology of MapReduce. Li J et al. [11] propose to outsource the key-issuing and the decryption to the cloud to reduce the burden of the mobile client device. Zhou Z, et al. [21] propose a privacy preserving CP-ABE scheme, in which the encryption and decryption are outsourced in the cloud. Lai J, et al. [22] introduce a semi-trusted proxy to modify a ciphertext under one access policy into ciphertexts but the proxy doesn't know the plaintext, which can be used to outsource the encryption to the cloud. Hohenberger S, et al [23] propose an offline/online scheme, in which the key generation and encryption are segmented two phase- the majority part is executed on offline and the rest is on inline. The scheme reduces the online encryption burden. But the schemes mentioned above have a drawback that the CSPs may compute the partial ciphertext honestly. So, the verifiable outsourcing ABE is proposed. Lai J et al. [13]

propose a variable homomorphic encryption. F. Armknecht et al. [12] introduce the conception of the outsourced proofs of retrievability (POR) and the user can use an external auditor to verify the POR. Parno B et. al [8] propose a verifiable outsourced CP-ABE. Ma H, et al. [24] propose two ciphertext-policy attribute-based key encapsulation mechanism (CP-AB-KEM) schemes that for the first time achieve both outsourced encryption and outsourced decryption in two system storage models. Li J, et al. [24] improve the algorithm of Green [7] which outsourced the decryption to the cloud and introduce the blinding algorithm to reduce the exponential computation of the encryption to a constant. Meanwhile, this scheme can verify the correctness of the outsourced decryption. Li J, et al. [25] propose a scheme in which not only the authorized users but also the unauthorized users can check the correctness of the outsourced decryption. Jiguo Li et al. in 2020 [26] propose verifiable outsourced decryption scheme which can simultaneously verifies the correctness for transformed ciphertext for the authorized users and unauthorized users. [27] firstly propose a imporved revocable CP-ABE scheme based chameleon hash function and apply it in IoT. Zheng, F, Peng, XG, et al. [28] propose an efficient and lightweight CP-ABE scheme by constructing an identical policy tree. Sowjanya K, Mou D, Ray S [29] imporved CP-ABE into a light-weight descrption scheme based ECC to make it suitable for mobile cloud service. Zhang, Z., W. Zhang, et al. [30] imporve hidden policy of CP-ABE to resist attribute values guess and use it to mobile cloud computing. Sarma R, Kumar C, et al. [31] advanced CP-ABE to suit for fog node in IoT by considering key-escrow resistance, revocable atribution, attribute addition, and outsourcing of expensive operations. Rs A, Cka B, et al. [32] propose a scheme of improving CP-ABE from multi-authority access control view for IoT. Yang L, Li C, et al. [33] improve CP-ABE by hiding both attribute values and names in policy by using private set intersection (PSI) to resist poly plaintext attacks. [34] propose CP-ABE exists secure debug in descryption. They propose a improved scheme based revocable idea.

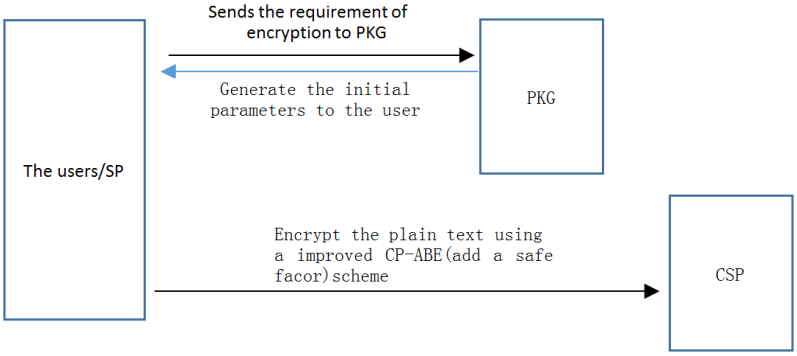

Figure 3 The process of encrytion with a improved CP-ABE scheme

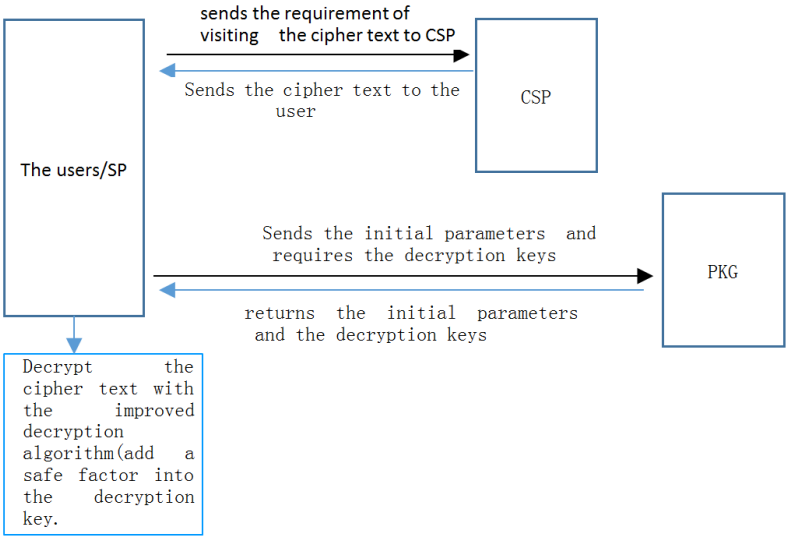

Figure 4 The process of decrytion with a improved decrption algorithm

But all these schemes from [1-33] are based on the model shown in Figure 1 and not in Figure 2. But in practice, the scenarios shown in Figure 2 exists widely. The existing schemes cannot maintain the security of the data if PKG and CSP collude by connecting the common field such as the telephone number of the user, by which PKG can provide the decription key for CSP. [34] involves in security breach, but it solves it from attribute revocation view.

In our paper, we apply our idea which is different from [34] to improve the existing scheme [25] and provide a secure scheme suitable for the second scenario.

3 Preliminary

3.1 Identifier Abbreviation

We list the abbreviation of the identifiers in Table 1.

Table 1 The abbreviation of Identifiers

Identifier abbreviation	description
CSP	Cloud Service Provider
SP	Service Provider
PKG	Primary Key Generation
MSK	Master Key
DTK	Decryption Transformation Key
ETK	Encryption Transformation Key
Γ	access structure
PK	public key
SK	private key
U	system attribute set
S	user's attribute set
g^{η}	safe factor

3.2 Bilinear Maps

Let $G_1$and G_2 as two groups of prime order p, g as a generator of group G_1 and $e(g,g)$ as a bilinear map of $G_1 \times G_1 \rightarrow G_2$. The following formula is true:

1) For $g \in G_1, a, b \in Z_p, e(g^a, g^b) = e(g,g)^{ab}$;

2) $e(g,g) = 1$, if and only if $g = 1$.

3.3 Decisional Bilinear Diffie-Hellman (BDH) Assumption

Let a, b, c, z are random numbers and $a, b, c, z \in Z_p$. Then no polynomial-time adversary can distinguish the tuple $(A = g^a, B = g^b, C = g^c, Z = e(g,g)^{abc})$ from the tuple $(A = g^a, B = g^b, C = g^c, Z = e(g,g)^z$ with more than a negligible advantage.

3.4 Linear Secret Sharing Scheme

In LSSS, let C as the set of participants, ϕ is monotone Boolean function and Ψ as access structure. There exists $\phi(B) = 1$ if and only if $A \in \Psi$. Let S as a secret, M as a $l \times n$ matrix on the finite field F_C and $v = (s, v_2, \ldots v_n)$ as a vector on F_C. Define $\lambda_i = vM_i$ and $T = (M, \phi)$. If we can find a $\omega_i \in F_p$ to make $\sum \omega_i \lambda_i = (1,0, \ldots 0)$; then $\sum_i \omega_i \lambda_i = s$ is true.

3.5 Li's Verifiable Outsourced ABE

Li's scheme [25] is improved based on Green's [7] scheme. The scheme outsources the encryption and decryption to CSP and the roles includes PKG, CSP and the user. The detailed steps are as follows:

-Setup: PKG takes the security parameter λ and a system attribute set $U = \{1,2 \ldots |U|\}$ as input, then outputs the public key $PK = \left(G_1, G_2, g, e(g,g)^{\alpha}, g^{\beta}, h_1, h_2, \ldots h_{|U|}, H\right)$ and the master key is $MSK = (\alpha, \beta)$.

-Key generation: PKG takes MSK and the user's attribute set $S \subset U$ as input, then outputs the private key $SK = z^{-1} \bmod p$ and the decryption transformation key $DTK = (K^z, K_0^z, \{K_i^z\})$ in which $K = g^{\alpha} g^{\beta t}, K_0 = g^t, K_i = h_i^t \ (i \in S)$. Random element $t, z \in Z_p$. Publish DTK.

-ETK generation: the user takes SK as input and outputs the encryption transformation key $ETK = (x, h_1^x, h_2^x, \ldots h_{|S|}^x)$. Keep secret ETK.

-Encryption: this step includes two steps, one of which is executed by the user and the other is outsourced to CSP.

(1) Partial ciphertext generation: the user takes PK, the message m and the access structure $\Gamma = (M, \phi)$ as input, then outputs partial ciphertext $(C_1 = e(g,g)^{\alpha s} \cdot m, C_2 = g^s, C_3 = g^{sr})(r = H(C_1, C_2, m), s, d \in Z_p)$ which is can be verified by $f(PK, C_1, m)$. If $f = 1$, then C_1 is correct. Otherwise, C_1 is computed wrongly. The another output is blinding factors $g^{\beta d} \cdot h_i^x$

-Outsourcing parameter generation: the user takes PK, ETK as input and outputs the outsourcing parameters $OP = \left(\lambda_i - d, sr - x, g^{\beta d} \cdot h_i^x\right)$ and uploads (C_1, C_2, C_3, OP) to CSP.

-Outsourcing: CSP takes OP as input and outputs the rest ciphertext:

$$\overline{C_1} = g^{\beta\lambda_i} \cdot h_i^{sr}$$

Then the ciphertext $C = (C_1, C_2, C_3, OP, \overline{C_1})$.

(2) Decryption: Decryption includes two steps, one of which is transformation and the other is final decryption.

(3) Transformation: CSP takes DTK and C as input and then outputs the transformation ciphertext C_T when $S \sqsubseteq \mathcal{T}$:

$$C_T = \frac{e(K^z, C_2)}{\prod_{i\in S}\left(e(K_0^z, \bar{C}_i)/e\left(K_i^z, C_3\right)\right)^{\omega_i}}$$

$$= \frac{e\left(g^{\alpha} g^{\beta t}, g^{s}\right)^{z}}{\prod_{i\in S}(e(g^t, g^{\beta\lambda_i} \cdot h_i^{sr})^z / e(h_i^t, g^{sr})^z)^{\omega_i}}$$

$$= \frac{e(g,g)^{\alpha sz} e(g,g)^{\beta tsz}}{e(g,g)^{zt\beta \sum \lambda_i \cdot \omega_i} \cdot \prod_i e(g,h_i)^{tsrz\omega_i} / e(h_i, g)^{-tsrz\omega_i}}$$

$$= \frac{e(g,g)^{\alpha sz} e(g,g)^{zt\beta s}}{e(g,g)^{zt\beta s}} = e(g,g)^{\alpha sz}$$

-Decryption: the user takes C_T and SK as inputs and outputs the plain text m if and only if the verification function $C_3 = C_2^{H(C_1, C_2, m)}$.

$$m = C_1 / C_T^{SK_S}$$

Li's scheme satisfies RCCA-secure.

4 The Overview of Our Model

To the individual, he/she often has the requirement of uploading his/her data to CSP by himself/herself without by SP, i.e. the platform that he/she often uses is the model shown in Figure 2. But the existing outsourced ABE scheme are all designed according to the model shown in Figure 1. For the model II, the existing schemes will result in the collusion between CSP and PKG by PKG transmitting the private key to CSP. So it's necessary to improve the exiting schemes to make it applicable to the senario in Figure 2.

4.1 The Steps of Our Model

We advanced Li's CP-ABE scheme and the content of our modle is as follows:

1. Setup. PKG takes the security parameter λ and a system attributes set U as input, then outputs the public key PK and the master secret key MSK. This step is the same as Li's scheme. Except the top step, PKG also computes the partial of a decryption transformation key DTK: p-DTK.
2. Key generation. The user inputs the master key MSK, a set $S \subseteq U$ of attributes and p-DTK. Then, gets a private key SK and the whole DTK. The step is executed by PKG in Li's scheme while it is done by the user in our model. Moreover, DTK in our scheme is changed compared with Li's scheme and added a safe factor g^{η}, the detailed realization will be described in section 5.
3. ETK generation. The user takes the set S of attributes and then computes its encryption transformation key ETK. This step is the same as Li's scheme.
4. Encryption. The encryption algorithm is deivided into the following three stages:
5. Partial ciphertext generation: The user takes the public key PK, a message m and an access structure $\Gamma = (M, \phi)$ as input, where the function ϕ associates rows of matrix M to the attributes. Then, it outputs partial ciphertext C_1 such that $f(PK, C_1, m) = 1$, where the output of function f is 0 or 1. In this step, C_1 is changed compared with Li's scheme for a safe factor g^{η} is added in C_1.
6. Outputsourcing parameter generation: the user inputs PK, ETK and computes the outsourcing parameters OP. This sep is the same as Li's scheme.
7. Outsourcing: The cloud server takes the outsourcing parameters OP as input and then outputs the rest part of the ciphertext C_2 based on the access structure. That is, the ciphertext is $C = (C_1, C_2)$. This step is also the same as Li's scheme and C_2 keeps the same as Li's scheme.
8. Transformation. The cloud server takes as input the DTK and ciphertext C, then returns the transformation ciphertext C_T if S satisfies the access structure Γ. In this step C_T is changed for C_1 is changed.
9. Decryption. The user takes as input the transformation ciphertext C_T and decryption key SK, then returns the message m if $f(PK, C_1, m) = 1$. Otherwise, it outputs ⊥.

4.2 Security Model

Our scheme is improved on the basis of Li's scheme whose security model obeys Green's RCCA-secure model [7]. Therefore, our security model is proved according to Green's, in which includes two roles: a simulator B and a

adversory $\mathcal{A}$. The secure model of our scheme is as follows:

1. Setup. B runs the algorithm of Setup and publish the public key PK to $\mathcal{A}$.
2. Phase 1. B constructs an empty table E and an empty table T. $\mathcal{A}$ sends continuous queries to B and the requirement is shown as follows:
3. Create (S). B takes a set of attributes S and executes the algorithm of Key generation. Then return (SK,DTK), fill (j,S,SK,DTK) into Table T and sends DTK to $\mathcal{A}$.
4. Corrupt(i). If the ith entry is involved in table T, B adds S into the table E and sends SK to $\mathcal{A}$. Otherwise, it will return $\perp$.
5. Decrypt(i,CT). If the ith entry is involed in table T, B will sends a message of CT to $\mathcal{A}$ by using the entry (i,S,SK,DTK). Otherwise, it returns $\perp$.
6. Challenge. $\mathcal{A}$ sends two messages m_0 and m_1 which have the equal length. In addition, $\mathcal{A}$ chooses an access structure Γ^* which does not satisfy the access structure Γ. By computing, B filps a coin b and sends the challenge ciphertext CT^* to $\mathcal{A}$. At the same time, $\mathcal{A}$ can visit the parameter OP^* related to the message CT^*.
7. Phase 2. Phase 1 is executed continuously under the following constraint:
8. A corrupt query for $S' \in \Gamma^*$;
9. Decryption queries for the cipher texts of both m_0 and m_1.
10. Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage of that the adversary $\mathcal{A}$ can gain is:

$$|\Pr[b' - b] - \frac{1}{2}|.$$

5 The Detailed Realization of Our Model

Before being improved, take Li's [25] scheme as example, $C_1 = e(g,g)^{\alpha s} \cdot m, C_2 = g^s$, in which g^s preserved on CSP. The adversary can obtain g^α from PKG. If PKG and CSP collude, they can compute $e(g,g)^{\alpha s}$ by:

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

then get the plain text $m = \frac{C_1}{e(g,g)^{\alpha s}}$.

On the principle of maintaining the most of the algorithm unchanged, we propose adding a factor g^η, in which $\eta \in Z_P$ and g^η is a random element generated by the user as a component of the whole algorithm. In the whole process of the advanced algorithm, only DTK and C_1 are changed compared with Li's scheme. The detailed steps are as follows:

1. Setup. this algorithm includes two steps, one of which is executed on PKG and the other is executed by the user:
2. PKG takes the security parameter λ and a system attribute set $U = \{1,2 \ldots |U|\}$ as input, then outputs the public key $PK = (G_1, G_2, g, e(g,g)^\alpha, g^\alpha, g^\beta, h_1, h_2, \ldots h_{|U|}, H)$ and the master key is $MSK = (\alpha, \beta)$. Then PKG continues to compute (K, K_0, K_i) in which $K = g^{\beta t}, K_0 = g^t, K_i = h_i^t$ $(i \in S, t \in Z_p)$.
3. the user generates random number $\eta, z \in Z_p$ and computes g^η. Keep η, z and g^η as a secret.
4. Key generation. The algorithm includes two steps: DTK generation and ETK generation. In Li's scheme [25], DTK generation is executed by PKG while in our scheme, the step is transfered to the user client.
5. DTK generation. The user computes $DTK = (K'^z, K_0'^z, \{K_i'^z\})$, in which $K' = (g^\alpha)^\eta g^{\beta t} (g^\beta)^\eta = g^{\alpha\eta} g^{\beta t} g^{\beta\eta}, K_0' = g^t \cdot g^\eta, K_i' = {h_i}^t \cdot g^\eta$ and publishes it, while in Li's scheme, $DTK = (K^z, K_0^z, \{K_i^z\})$ and $K = g^\alpha g^{\beta t}, K_0 = g^t, K_i = h_i^t$ $(i \in S)$.
6. ETK generation: This step Keeps Li's scheme [25]. In this step we can get the secret encryption transformation key $ETK = (x, h_1^x, \ldots, h_{|S|}^x)$ which is only visibles to the user. This step is also executed by the user.
7. Encryption. in general, this step keeps the large content of Li's scheme [25] and only C_1 has been changed. The process is still divided into three steps, stage I and stage II of which are executed by the user and stage III is executed by CSP. Then finally the ciphertext is as follows:

$$C = (C_1, C_2, C_3, \overline{C_i})$$

In which $(C_1 = (e(g,g)^\alpha)^{\eta s} \cdot m, C_2 = g^s, C_3 = g^{sr}, \overline{C_i} = g^{\beta\lambda_i} \cdot h_i^{sr})$.

8. Decryption. the decryption algorithm includes two steps, one of which is transformation executed by CSP and the other is final decryption executed by

the user.

9. Transformation: CSP takes DTK and C as input and then outputs the transformation ciphertext C_T when $S \sqsubseteq \mathcal{T}$:

$$C_T = \frac{e(K'^z, C_2)}{\prod_{i\in S}\left(e(K_0'^z, \bar{C}_i)/e(K_i'^z, C_3)\right)^{\omega_i}} = \frac{e\left(g^{\alpha\eta}g^{\beta t}\cdot g^{\beta\eta}, g^s\right)^z}{\prod_{i\in S}(e(g^t\cdot g^{\eta}, g^{\beta\lambda_i}\cdot h_i^{sr})^z / e(h_i^t\cdot g^{\eta}, g^{sr})^z)^{\omega_i}}$$

$$= \frac{e(g,g)^{\alpha sz\eta}e(g,g)^{\beta tsz}e(g,g)^{\beta\eta sz}}{e(g,g)^{zt\beta\sum\lambda_i\cdot\omega_i}\cdot e(g,g)^{z\eta\beta\sum\lambda_i\cdot\omega_i}\cdot\prod_i e(g,h_i)^{tsrz\omega_i}/e(h_i,g)^{tsrz\omega_i}\cdot\prod_i e(g,h_i)^{\eta srz\omega_i}/e(h_i,g)^{\eta srz\omega_i}}$$

$$= \frac{e(g,g)^{\alpha sz\eta}e(g,g)^{zt\beta s}e(g,g)^{\beta\eta sz}}{e(g,g)^{zt\beta s}\cdot e(g,g)^{z\eta\beta s}} = \mathrm{e(g,g)}^{\alpha s\eta z}$$

10. Decryption. this step keeps Li's scheme [25]. z^{-1} denotes the secret key. The plain text and the verification function are as follows:

$$\mathrm{m} = \frac{C_1}{C_T^{SK}} = \frac{e(g,g)^{\alpha s\eta}\cdot m}{(e(g,g)^{\alpha s\eta z})^{z^{-1}}} = \frac{e(g,g)^{\alpha s\eta}\cdot m}{e(g,g)^{\alpha s\eta}}$$

$$C_3 = C_2^{H(C_1,C_2,m)}$$

6 Security

Li [25] have proved their scheme satisfied RCCA-secure.

1. Theorem 1. Suppose decisional BDH assumption holds, then no polytime adversary can selectively break our system.
2. Theorem 2. Suppose Green's scheme [7] satisfies CPA-security, our scheme also satisfies CPA-security.
3. Suppose $\mathcal{A}$ as the adversary which has non-negligible advantage ϵ to break our system and B as a simulator that is an algorithm and can solve the BDH problem with advantage ϵ. Then the game is as follows:
4. Init. Suppose $\alpha\eta = \alpha'\eta + ab\eta$.
5. Setup. The system issues the parameters: $(\mathrm{G_1, G_2, g, e(g,g)^{\alpha}, g^a, g^{\beta}, h_1, h_2, \ldots h_{|U|}, H, \alpha, \beta, K, K_0, K_i})$, $\mathrm{K = g^{\beta t}, K_0 = g^t, K_i = h_i^t}$ $(\mathrm{i \in S, t \in Z_p})$.
6. Query phase 1. 1) Secret key query. $\mathcal{A}$ constructs many attributes set $S1, S2, S3 \ldots$ to issue corresponding requests of secret keys to B. B generates $K_i1, K_i2, K_i3 \ldots$ to B. Then B sends them to $\mathcal{A}$.

Generate transformation key. B randomly selects $\mathrm{z, \eta} \in Z_p$ and then computes $\mathrm{DTK}_j =< K'^z, K_0'^z, K_i\mathrm{j}'^z >$ and $\mathrm{ETK}_j =< x, h_1^x, h_2^x, \ldots, h_{Sj}^x >$. Then B generates a Table T and adds $< \mathrm{j++, Sj, K}ij, \mathrm{DTK}_j, ETK_j >$ into T.

Encrypt M^*. $\mathcal{A}$ constructs $<M^*, Si>$ to B and B randomly select $\mathrm{s}, \lambda_j \in Z_p$, then calls Encryption to get $\mathrm{CT^* = (C_1, C_2, C_3, \bar{C}_1)}$.

7. Challenge. $\mathcal{A}$ sends two equal length messages $< \mathrm{M0}, \Psi >$ and $< \mathrm{M1}, \Psi >$ to B. In these messages, Si constructed in query phase 1 doesn't satisfy Ψ. B randomly selects $\beta \in (0,1)$ and encrypts M_β as CT^* by calling the step o f 3) in Query phase 1..
8. Query phase 2. Run Query phase 1, but Si constructed in query phase 1 doesn't satisfy Ψ.
9. Guess. $\mathcal{A}$ outputs $\beta' \in \{0,1\}$ of β to B. If $\beta = \beta'$, B outputs $\mu = 0$ and guesses that $\mathrm{Y = e(g,g)}^{ab\eta s}$; otherwise, B outputs $\mu = 1$ and recognizes that Y is a random element.

If $\mu = 1$, $\mathcal{A}$ will regard Y as a random element. Then at this moment, the following formula:

$$\Pr\{\mathrm{B}(\bar{y}, Y = R) = 0\} = \Pr\{\beta' \neq \beta | \mu = 1\} = \Pr\{\beta' = \beta | \mu = 1\} = \frac{1}{2}$$

is valid. If $\mu = 0$, $\Pr\{\mathrm{B}(\bar{y}, Y = e(g,g)^{abs\eta}) = 0\} = \Pr\{\beta = \beta' | \mu = 0\} = \frac{1}{2} + \epsilon$

Therefore, the overall advantage of B is as follows:

$$\frac{1}{2}\Pr\{B(\bar{y}, Y = e(g,g)^{abs\eta}) = 0\} + \frac{1}{2}\Pr\{B(\bar{y}, Y = R) = 0\} - \frac{1}{2} = \frac{1}{2}\times\frac{1}{2} + \frac{1}{2}\times\left(\epsilon + \frac{1}{2}\right) - \frac{1}{2} = \frac{1}{2}\epsilon < \epsilon$$

The proof is complete.

7 Comparison and Analysis

In our improved scheme, there are many computation on the user's device such as $\mathrm{K' = (g^{\alpha})^{\eta} g^{\beta t} (g^{\beta})^{\eta} = g^{\alpha\eta} g^{\beta t} g^{\beta\eta}, K_0'^z = g^t \cdot g^{\eta}, K_i'^z = h_i{}^t \cdot g^{\eta}}$, etc. But these operation are all power series which cannot occupy a lot of time.

7.1 Comparison of Executing Location

The executing location of every step of this algorithm is

shown in Table 2 when comaring with Li's scheme.

Table 2 Comparison of executing steps between Li's scheme and our scheme

	Setup	**Key generation (SK)**	**Key generation (DTK)**	**ETK generation**	**Encryption (stage I)**	**Encryption (stage II)**	**Encryption (stage III)**	**Transformation**	**Decryption**
Li's scheme	PKG	SP	PKG	The user	The user	The user	CSP	CSP	The user
Our scheme	PKG	The user	PKG+the user	The user	The user	The user	CSP	CSP	The user

From Table 2, we can see that the executer of the steps of both our scheme and Li's scheme are idential largely except in the step of Key generation (SK) and Key generation (DTK). But the main difference of the two is whether the relationship between PKG and the user is trusted. In our scheme the relationship of the user and PKG is untrusted while in Li's it is trusted. In Li's scheme, PKG, SP and the user trust each other and their relationship satisfies the architecture in Figure 1, while in our scheme, PKG and the user do not trust each other and the relationship of them satisfies the architecture in Figure 2.

7.2 Comparison of Change

The changable part of our scheme compared with Li's scheme is shown in Table 3.

Table 3 Comparison of changable part between Li's scheme and our scheme

Setup	**Key generation**	**ETK generation**	**Encryption (stage I)**	**Encryption (stage II)**	**Encryption (stage III)**	**Transformation**	**Decryption**
√	DTK	√	C_1	√	√	C_T	√

√ denots that it has no change in the step of our scheme compared with in that of Li's scheme. From Table 3, we can see that DTK, C_1 and C_T changed. Both DTK and C_1 are respectively added the safe factor g^{η} and C_T changed with the change of C_1.

7.3 Comparison of Computing Cost

The detailed computation cost of both Li's scheme and ours is shown as Table 4. t_i denotes running time in the ith step. O denotes the increased computaion cost based on Li's scheme.

Table 4 Computation cost between Li's scheme and our scheme

	Setup	**Key generation (SK)**	**Key generation (DTK)**	**ETK generation**	**Encryption (stage I)**	**Encryption (stage II)**	**Encryption (stage III)**	**Transformation**	**Decryption**
Li's scheme	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
Our scheme	t_1	t_2	$t_3 \times O(e^n)$	t_4	$t_5 \times O(e^n)$	t_6	t_7	t_8	t_9

Form Table 4, we can see in step of Key generation (DTK), the running time changed to $t_3 \times O(e^n)$, the reason of which is that $K' = (g^{\alpha})^{\eta} g^{\beta t} (g^{\beta})^{\eta} = g^{\alpha\eta} g^{\beta t} g^{\beta\eta}, K_0'^{z} = g^{t} \cdot g^{\eta}, K_i'^{z} = {h_i}^{t} \cdot g^{\eta}$ in our schme, while $K = g^{\alpha} g^{\beta t}, K_0 = g^{t}, K_i = h_i^{t}$ in Li's scheme. The increased computaional cost in our scheme is $(g^{\alpha})^{\eta}$. Suppose g^{α} is a constant, then the computation cost of $(g^{\alpha})^{\eta}$ is $o(e^n)$. Similarily, the computation cost of the step Encryption (stage I) is also so.

8 Conclusion and Outlook

In this paper, we propose a scheme to protect the security in the scenarios where the user interact with CSP directly and PKG is shared by all the users and may not be trusted completely by adding a factor into the ciphertext and private keys. We apply the idea into Li's scheme which is a algorithm to provide a safe outsourcing

service. Finally, we prove the improved scheme secure.

We think our idea can apply to improve other existing algorithm and eager to the new ideas and algorithms to be provided for the scenario in Figure 2.

In this paper, we realize our idea by modifing Li's scheme, which just an small example. We hope it is an enlightenment to apply our idea into other outstanding CP-ABE schemes and create many better schemes for the architecture of Figure 2.

References

[1] Sahai A, Waters B. Fuzzy Identity-Based Encryption [J]. 2005.

[2] Goyal V, Pandey O, Sahai A, et al. Attribute-based encryption for fine-grained access control of encrypted data [C] // Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006. ACM, 2006.

[3] Bethencourt J, Sahai A, Waters B. Ciphertext-Policy Attribute-Based Encryption [C] // IEEE Symposium on Security & Privacy. IEEE, 2007.

[4] Waters B. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization [C] // Public Key Cryptography-pkc -international Conference on Practice & Theory in Public Key Cryptography. Springer, Berlin, Heidelberg, 2011.

[5] Soliman H. Securing communication data in pervasive social networking based on trust with KP-ABE [J]. Computing reviews, 2022(7): 63.

[6] Das S, Namasudra S. MACPABE: Multi-Authority-based CP-ABE with efficient attribute revocation for IoT-enabled healthcare infrastructure [J]. International journal of network management, 2023.

[7] Green M, Hohenberger S, Waters B. Outsourcing the decryption of ABE ciphertexts [C] // Proceedings of the 20th USENIX conference on Security. USENIX Association, 2011.

[8] Parno B, Raykova M, Vaikuntanathan V. How to Delegate and Verify in Public: Verifiable Computation from Attribute-Based Encryption [C] // International Conference on Theory of Cryptography. Springer-Verlag, 2012.

[9] Frikken K, Atallah M, Li J. Attribute-Based Access Control with Hidden Policies and Hidden Credentials [J]. IEEE Transactions on Computers, 2006, 55(10): 1259-1270.

[10] Chim T W, Yuen T H. Outsourcing Encryption of Attribute-Based Encryption with MapReduce [J]. 2012, 10.1007/978-3-642-34129-8(Chapter 17): 191-201.

[11] Li J, Chen X, Li J, et al. Fine-Grained Access Control System Based on Outsourced Attribute-Based Encryption [J]. 2013.

[12] F. Armknecht, J. Bohli, G. O. Karame, Z. Liu, and C. A. Reuter, "Outsourced proofs of retrievability," in Proc. ACM SIGSAC Conf. Comput. Commun. Security, 2014, pp. 831–843.

[13] Lai J, Deng R H, Pang H, et al. Verifiable Computation on Outsourced Encrypted Data [J]. 2014.

[14] Song, Park. Attribute Based Proxy Re-encryption for Data Confidentiality in Cloud Computing Environments [C] // Acis/jnu International Conference on Computers. IEEE Computer Society, 2011.

[15] Li Q, Ma J, Li R, et al. Large universe decentralized key-policy attribute-based encryption [J]. Security & Communication Networks, 2015, 8(3): 501-509.

[16] Qi, Li, Jianfeng, et al. Large universe decentralized key-policy attribute-based encryption [J]. Security & Communication Networks, 2014.

[17] Lai J, Deng R H, Li Y. Fully Secure Cipertext-Policy Hiding CP-ABE [J]. 2011.

[18] Liu Z, Cao Z, Wong D S. Traceable CP-ABE: How to Trace Decryption Devices Found in the Wild [J]. IEEE Transactions on Information Forensics & Security, 2017, 10(1): 55-68.

[19] Odelu V, Das A K, Rao Y S, et al. Pairing-based CP-ABE with constant-size ciphertexts and secret keys for cloud environment [J]. Computer Standards & Interfaces, 2016, 54(PT.1): 3-9.

[20] Yan X, He G, Yu J, et al. Offline/Online Outsourced Attribute-Based Encryption with Partial Policy Hidden for the Internet of Things [J]. Journal of Sensors, 2020, 2020(7): 1-11. DOI: 10.1155/2020/8861114.

[21] Zhou Z, Huang D. Efficient and secure data storage operations for mobile cloud computing [C] // International Conference on Network & Service Management. IEEE, 2012.

[22] Lai J, Deng R H, Yang Y, et al. Adaptable Ciphertext-Policy Attribute-Based Encryption [C] // International Conference on Pairing-Based Cryptography. Springer International Publishing, 2013.

[23] Hohenberger S, Waters B. Online/Offline Attribute-Based Encryption [C] // International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2014.

[24] Ma H, Zhang R, Wan Z, et al. Verifiable and Exculpable Outsourced Attribute-Based Encryption for Access Control in Cloud Computing [J]. IEEE Transactions on Dependable & Secure Computing, 2017, 14(6): 679-692.

[25] Li J, Li X, Wang L, et al. Fuzzy encryption in cloud computation: efficient verifiable outsourced attribute-based encryption [J]. Soft Computing, 2017.

[26] Li J, Wang Y, Zhang Y, et al. Full Verifiability for Outsourced Decryption in Attribute Based Encryption [J]. IEEE Transactions on Services Computing, 2020, 1939: 1.

[27] Guo R, Yang G, Shi H, et al. O-R-CP-ABE: An Efficient and Revocable Attribute-Based Encryption Scheme in the Cloud-Assisted IoMT System [J]. IEEE Internet of Things Journal, 2021, PP (99): 1.

[28] Zheng, F, Peng, XG, Li, ZD, et al. An Efficient User's Attribute Revocation Scheme Suitable for Data Outsourcing in Cloud Storage [J]. WIRELESS COMMUNICATIONS & MOBILE COMPUTING, 2022.

[29] Sowjanya K, Mou D, Ray S. A lightweight key management scheme for key-escrow-free ECC-based CP-ABE for IoT healthcare systems [J]. Journal of Systems Architecture, 2021, 117(2): 102108.

[30] Zhang, Z., W. Zhang, and Z. Qin. "A partially hidden policy CP-ABE scheme against attribute values guessing attacks with online privacy-protective decryption testing in IoT assisted cloud computing." Future Generation Computer Systems 123(2021).

[31] Sarma R, Kumar C, Barbhuiya F A. PAC-FIT: An efficient privacy preserving access control scheme for fog-enabled IoT [J]. Sustainable Computing: Informatics and Systems, 2021, 30(2): 100527.

[32] Rs A, Cka B, Fab A. MACFI: A multi-authority access control scheme with efficient ciphertext and secret key size for fog-enhanced IoT [J]. 2021.

[33] Yang L, Li C, Cheng Y, et al. Achieving privacy-preserving sensitive attributes for large universe based on private set intersection [J]. Information Sciences, 2022, 582: 529-546.

[34] Ning J, Cao Z, Dong X, et al. CryptCloud: Secure and Expressive Data Access Control for Cloud Storage [J]. IEEE Transactions on Services Computing, 2021, 14(1): 111-124.

Biography

Fang Zheng

Born in 1982, Master. Her research area: information security, network security, privacy protection.

E-mail: alice20110324@126.com

Pingzhen Li

Born in 1963, Master. Her research area: information security

E-mail: lipz@sxufe.edu.cn

Xinguang Peng

Born in 1958, Professor, Ph.D. His research area: information security, network security.

E-mail: sxgrant@126.com

Zhidan Li

Born in 1989, Ph.D. His research area: information security, network security.

E-mail: zhidanli@bupt.cn