Multimedia Tools and Applications (2021) 80:31487–31508
https://doi.org/10.1007/s11042-021-11286-0

1194: SECURED AND EFFICIENT CONVERGENCE OF ARTIFICIAL INTELLIGENCE AND INTERNET OF THINGS

Efficient access control with traceability and user revocation in IoT

Yi Wu[1] · Wei Zhang[1] · Hu Xiong[1] · Zhiguang Qin[1] · Kuo-Hui Yeh[2]

Received: 15 October 2020 / Revised: 7 May 2021 / Accepted: 8 July 2021 /
Published online: 10 September 2021

Abstract
With the universality and availability of Internet of Things (IoT), data privacy protection in IoT has become a hot issue. As a branch of attribute-based encryption (ABE), ciphertext policy attribute-based encryption (CP-ABE) is widely used in IoT to offer flexible one-to-many encryption. However, in IoT, different mobile devices share messages collected, transmission of large amounts of data brings huge burdens to mobile devices. Efficiency is a bottleneck which restricts the wide application and adoption of CP-ABE in Internet of things. Besides, the decryption key in CP-ABE is shared by multiple users with the same attribute, once the key disclosure occurs, it is non-trivial for the system to tell who maliciously leaked the key. Moreover, if the malicious mobile device is not revoked in time, more security threats will be brought to the system. These problems hinder the application of CP-ABE in IoT. Motivated by the actual need, a scheme called traceable and revocable ciphertext policy attribute-based encryption scheme with constant-size ciphertext and key is proposed in this paper. Compared with the existing schemes, our proposed scheme has the following advantages: (1) Malicious users can be traced; (2) Users exiting the system and misbehaving users are revoked in time, so that they no longer have access to the encrypted data stored in the cloud server; (3) Constant-size ciphertext and key not only improve the efficiency of transmission, but also greatly reduce the time spent on decryption operation; (4) The storage overhead for traceability is constant. Finally, the formal security proof and experiment has been conducted to demonstrate the feasibility of our scheme.

Keywords IoT · Attribute-based encryption · Constant-size ciphertext and key · Traceability · User revocation

1 Introduction

With the rapid development of network and smart devices, Internet of Things (IoT) has penetrated into people's daily life, such as smart cars, smartphones, wearable devices and industrial Internet of Things. In the IoT, mobile devices are connected through the Internet

✉ Wei Zhang
zhangving@uestc.edu.cn

Extended author information available on the last page of the article

and exchange data with each other. With the development of the IoT, privacy-preserving has become the focus of attention, so the secure information exchange between mobile devices determines the smooth implementation of the Internet of things system. However, most mobile devices are resource-constrained, the storage and processing of massive information data brings heavy expenses to mobile devices. Fortunately, cloud computing can provide reliable computing services for most users, regardless of time and place. With the help of cloud computing, a large amount of data collected is outsourced to the cloud. On the other hand, for mobile devices that exit the system, their system permissions need to be reclaimed, so that the users who exit the system no longer have access to the system. In addition, there are some situations that we need to pay attention to. Some mobile devices disclose their keys for profit, which is likely to cause unauthorized users to access the system, so it is necessary to trace the subject of the leak. Therefore, there is a growing need to design an efficient CP-ABE scheme that supports user revocation and user traceability for the IoT.

Public key encryption is extensively regarded as one of the core technologies to prevent user privacy from being disclosed, but the traditional public key encryption system can only achieve one-to-one encryption. One-to-many encryption is implemented in the ABE scheme. The existing ABE schemes are divided into key policy attribute-based encryption (KP-ABE) [8] and ciphertext policy attribute-based encryption (CP-ABE) [3]. Among them, KP-ABE associates the access policy with the user's private key and the user attribute with the ciphertext. On the contrary, CP-ABE associates the access policy with the ciphertext, and the user attribute with the user's private key. Because CP-ABE is more in line with the actual application scenario, CP-ABE has attracted more attention from the industry and academic community. Taking an IoT based CP-ABE as an example, cloud service provider **A** stores large amounts of encrypted data. All users in the system can download the encrypted data on the cloud server. However, only users whose attributes satisfy the access policy in ciphertext can decrypt the data. A detailed instance is shown in Fig. 1. The data owner stores the encrypted data on cloud service provider **A**. The attribute set of the data user is {Name: "Bob", Age: 22, Identity: "student", Gender: "male"}. The access policy made by the data owner is (Identity: "student") AND (Gender: "male"). Because Bob satisfies the condition that the identity is a student and the gender is male, Bob can access the ciphertext and perform decryption operations. Data users who have the right to decrypt can download and decrypt the ciphertext from **A**. And, in CP-ABE, different users can have the same decryption rights as long as they have the same attributes. Some users in the system disclose their private keys for profit without the risk of being caught. In the face of the temptation of economic interests and "0" risk, many users are willing to disclose their decryption keys. This illegal disclosure of the key seriously threatens the privacy of data owners and system security. Therefore, it is necessary to implement a traceable CP-ABE scheme. In order to ensure that the caught malicious user no longer attack the security of the system, malicious users should be revoked in time. Once the revocation occurs, the revoked user can no longer access the system data. And due to the update of the ciphertext, even if the unauthorized user has an illegally obtained key, s/he cannot successfully decrypt the ciphertext. Moreover, in most existing ABE systems, the ciphertext and the key will grow with the increase of the corresponding attributes. The increasing ciphertext and key length have caused a huge computing burden for users, especially for mobile device users. The above problems hinder the wide application of CP-ABE encryption mechanism in IoT.

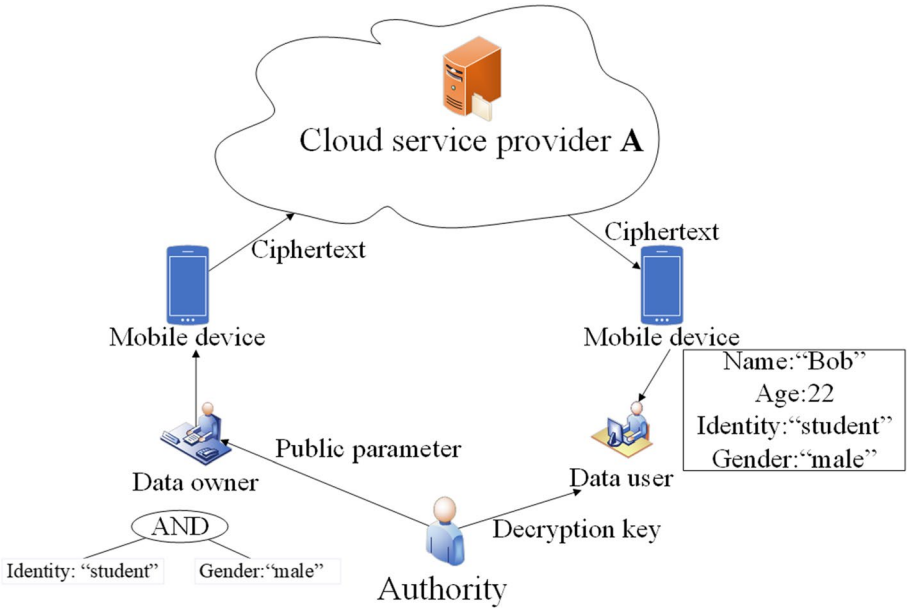

Fig. 1 An IoT system based on CP-ABE

1.1 Related work

In order to achieve flexible access control and assure data confidentiality, Sahai and Water [18] proposed the first attribute-based encryption scheme. Later, inspired by this scheme [18], some scholars carried out a series of related research on attribute-based encryption(ABE). Because ABE implements fine-grained access control, it is considered to be one of the most promising cryptographic primitives for flexible and secure data sharing. Because CP-ABE is more in line with the actual application scenario, CP-ABE has attracted more attention from the industry and academic community. Nonetheless, there are three obstacles that limit the research and practical application of traditional ABE, including efficiency, key abuse and user revocation.

The computing time of encryption and decryption in most CP-ABE scheme will increase with the number of access policy related to ciphertext in the system, which requires huge storage space and computation resource for mobile devices with limited capability. Some existing schemes have focused on this problem and given solutions. Outsourcing computing is one of the ways to solve this problem. Considering the decryption burden of data users, Green et al. [9] proposed an outsourced decryption ABE scheme. In addition to outsourced computing, constant-size ciphertext and key are also essential to improve the efficiency of CP-ABE schemes. In the traditional CP-ABE scheme, the size of ciphertext and key will increase with the increase of the number of attributes, resulting in a sharp decline in the performance of the system. Therefore, how to control the size of ciphertext and key is an urgent problem to be solved. Some schemes [2, 6, 7, 23] implement constant-size ciphertext, while others [10] implement constant-size key. For those who support both constant-size ciphertext and key, there has been a lot of research here [15, 16]. Among them, the constant-size ciphertext and

key scheme [15] proposed by Odelu solves the efficiency problem in IoT, but neither of them solved the problems of key abuse and user revocation.

Key abuse is another common problem in CP-ABE. For a CP-ABE scheme, as long as the user's attributes meet the access policy in the ciphertext, the user can perform the decryption operation. Therefore, there may be multiple users with access to the same ciphertext. Once the key disclosure occurs, the system will not be able to identify who leaked the key. Because users have no risk of being caught, users often sell their keys for economic gain in CP-ABE. In order to solve the above problem, Li et al. [12] proposed the first key accountability ABE scheme, but the access policy of this scheme can only be expressed as "AND gate and wildcard". In order to achieve stronger expression ability, Liu et al. [13] and Zhen et al. [24] successively implemented white-box traceability ABE and black-box traceability ABE which support arbitrary monotonous access structure. White-box traceability and black-box traceability are the two existing traceability methods. If the user in the system deliberately leaks his decryption key to the unauthorized user, then the system traces back to the malicious user according to the information contained in the key, which is called white-box traceability. Besides, if the user leaks the decryption device rather than the decryption key, the system finds out the identity of the user according to the device, which is called black-box traceability. To put it simply, the biggest difference between white-box traceability and black-box traceability is whether the malicious user leaks the decryption key or the decryption device. Li et al. [11] proposed a CP-ABE scheme with multiple authorities and accountable malicious users for the first time, but the access policy of this scheme can only be expressed as "AND" gate. After that, Ning et al. [14] proposed a white-box accountable CP-ABE scheme and the storage overhead for traitor tracing is constant.

In addition, for an ABE system, there are often some problems, such as user exit, traitor revoke and so on. Therefore, revocation is a problem that a perfect ABE system has to solve. Peng et al. [17] proposed a scheme to realize user revocation without redistributing the key or re-encrypting all the ciphertext. In this scheme, the user revocation is realized by integrating the unique identity of the revocation user in the ciphertext. Attrapadung et al. [1] utilize the linear secret sharing technique and identity-based multicast encryption technology [4] to propose two ABE revocation modes: direct revocation and indirect revocation. At present, most schemes adopt the way of indirect revocation, which is carried out by the authority, and the attribute revocation is realized by changing the key periodically. Indirect revocation does not need to introduce the revocation list, and the operation is flexible, but its disadvantage is that the revocation cost is high. On the other hand, the direct revocation is performed by the data owner, and the revocation list is embedded in the ciphertext to realize the revocation. Xu and Martin [19] proposed a dynamic user revocation scheme, which uses the cloud server to re-encrypt the ciphertext, and the cloud server updates the user revocation list. Once the user is added to the user revocation list, the user will lose all access to the system. Zhang et al. [22] proposed a CP-ABE scheme to realize both attribute revocation and user revocation. This scheme introduces an auxiliary function to specify the ciphertext related to revocation, and then uses the broadcast encryption technology to update only the ciphertext specified by the auxiliary function, so as to realize the attribute revocation and the direct revocation of the user.

From what has been discussed above, it can be seen that constant-size ciphertext and keys, traceability and revocability are critical to CP-ABE, but unfortunately, as far as we know, there is no solution that implements all three properties at the same time.

1.2 Our contribution

In this article, we focus on three common problems of CP-ABE applications in IoT: efficiency, key abuse and user revocation. In order to solve the three problems mentioned above, we propose an efficient CP-ABE scheme with constant-size ciphertext and key that supports user traceability and user revocation in IoT.

(1) **High efficiency**. We utilize the constant-size ciphertext and key scheme constructed by Odelu et al. [16] to improve the efficiency of this scheme, so as to maintain the local computing overhead in the decryption phase at a lightweight constant value. It not only improves the communication efficiency but also saves the decryption time.
(2) **White-box traceability**. Shamir's $(\bar{t}, \bar{n})$ threshold scheme is used to achieve white-box traceability. It can track those users who maliciously disclose their decryption keys, so as to solve the problem of key abuse. And our scheme only needs to store $\bar{t} - 1$ points on the polynomial $f(x)$ and the value $f(0)$ without maintaining a user table. The storage overhead of the scheme for tracing is constant.
(3) **User revocation**. For the revocation of malicious users, our scheme only requires the third-party server to update the ciphertext when it receives the revocation signal, and does not need the user to update the key periodically. Once the user is revoked, the user can no longer access to all the data in the system.
(4) **Secure and experimental analysis**. What's more, in this paper, we prove that our proposed scheme is chosen ciphertext attack (CCA) secure. Finally, we compare with other related schemes from the two aspects of theoretical analysis and simulation experiments. The results show that our scheme is not only functional but also efficient.

2 Preliminary

In this section, we list the mathematical primitives and basic definitions related to our scheme.

2.1 Access structure and attribute

We define the AND gate access structure similar to the method in [10] and [16]. The access structure is represented by attributes in the attribute field $\mathbb{A} = \{A_1, A_2, \ldots, A_n\}$. $S \in \mathbb{A}$ is a user's attribute set. And S is an n-byte string such as $c_1 c_2 \ldots_n$. The definition is as follows:

$$\begin{cases} c_i = 0, A_i \notin S \\ c_i = 1, A_i \in S \end{cases} \tag{1}$$

Give the following specific example to explain. If $n = 5$ and a user's attribute string is $S = c_1c_2c_3c_4c_5 = 10101$, then this means that the user has attributes A_1, A_3 and A_5. $|S|$ is used to represent the number of attributes in S.

P is used to represent the access policy made by the data owner. And the access policy is an n-byte string $b_1b_2 \dots b_n$. The definition is as follows:

$$\begin{cases} b_i = 0, A_i \notin P \\ b_i = 1, A_i \in P \end{cases} \tag{2}$$

Similarly, we give a specific column to further illustrate the above definition. If $n = 5$ and specified access policy is $P = b_1b_2b_3b_4b_5 = 11101$, then this means that the access policy includes attributes A_1, A_2, A_3 and A_5. $|P|$ is used to represent the number of attributes in P.

$P \in S$ is equivalent to having $c_i > b_i$ for any $i = 1, 2, \dots, n$. If $P \in S$, we say that the attribute set S satisfies the access policy P.

2.2 Bilinear group

Given two multiplicative cyclic groups, G, G_T. p is the prime order of G and G_T. g is a generator of G. The bilinear group $G \times G \to G_T$ satisfies the following properties [?].

(a) Non-degeneracy: $e(g, g) \neq 1$.

(b) Bilinearity: $\forall\, x, y \in G, c, d \in Z_p$, the equation $e(x^c, y^d) = e(x, y)^{cd}$ is true.

(c) Computability: the map e: $G \times G \to G_T$ can be effectively calculated.

2.3 Lagrange Interpolation Theorem

Let p be a prime, $f(x)$ be a k-order polynomial, $j_0, \dots, j_k$ be different elements in the field of Z_p integers, and $f_0 = f(j_0), \dots, f_k = f(j_k)$. By using the Lagrange interpolation theorem, the polynomial $f(x)$ can be expressed as:

$$f(x) \triangleq \sum_{i=0}^{k} \left(f_i r_i(x)\right)$$

where

$$r_i(x) \triangleq \prod_{t=0, t\neq i}^{k} \frac{j_t - x}{j_i - j_t}$$

is the Lagrange coefficient.

2.4 Shamir $(\bar{t}, \bar{n})$ threshold scheme

Shamir threshold scheme $(\bar{t}, \bar{n})$ is widely used in cryptography. The scheme utilizes the following principle: $\bar{t}$ points on the $\bar{t} - 1$ order curve can determine the curve, that is, $\bar{t}$ points are sufficient to determine a $\bar{t} - 1$ order polynomial. For a $(\bar{t}, \bar{n})$ threshold scheme,

a secret can be divided into $\bar{n}$ parts (or more), each of which is distributed to members as a separate part. All members can restore the secret of sharing. Suppose the secret is an element over a finite field Z_p^*. Randomly select $\bar{t}-1$ coefficients $a_1, a_2, \ldots, a_{\bar{t}-2}, a_{\bar{t}-1} \in Z_p^*$ and set the secret to constant a_0. Notice that we have the following polynomials: $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{\bar{t}-1} x^{\bar{t}-1}$. Each member is given a point on the curve (x, y), where x is the input of the polynomial and $y = f(x)$ is its corresponding output. Given any $\bar{t}$ points, we can restore the constant a_0 by Lagrange Interpolation Polynomial.

2.5 *l*-SDH assumption

G is a bilinear group with generator g and p is the order of G, the l-SDH (l-Strong Diffie-Hellman) problem in G is defined as below: provided a $(l+1)$-tuple $(g, g^y, g^{y^2}, \ldots, g^{y^l})$ as input, output a pair $(c, g^{\frac{1}{y+c}}) \in \mathbb{Z}_p \times G$. $\mathcal{A}$ has advantage λ in breaking this assumption if

$$Pr[\mathcal{A}(g, g^y, g^{y^2}, \ldots, g^{y^l}) = (c, g^{\frac{1}{y+c}})] \geqslant \lambda$$

where the probability is over the random bits employed by $\mathcal{A}$ and the random option of y in $\mathbb{Z}_p$.

Definition 1 In G, if no t-time $\mathcal{A}$ has advantage of greater than λ to break this assumption, we say that (l, t, λ)-SDH assumption stand in G.

3 System model and security model

In this section, we will introduce the entities contained in our system, as well as the system model, security model, and traceability model.

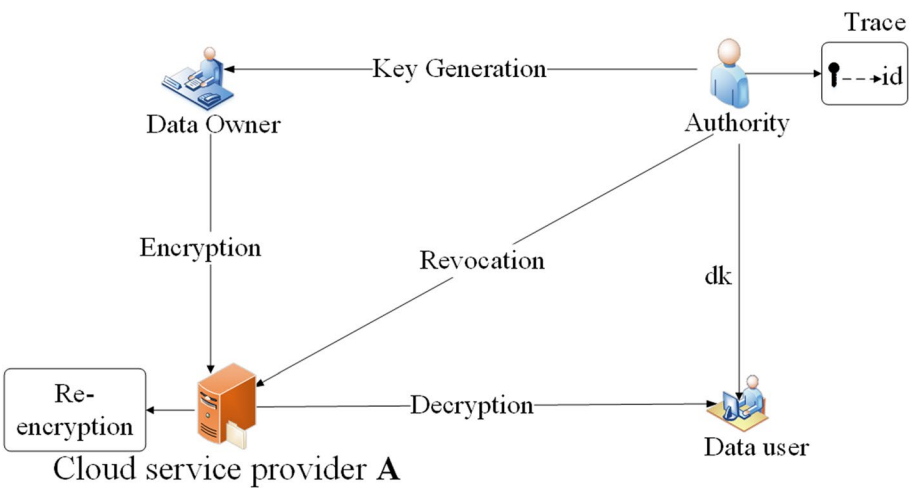

Fig. 2 system model

3.1 Entities in the system

In our newly proposed IoT system based on CP-ABE, there are four entities involved, namely, authority (*AT*), data user (*DU*), data owner (*DO*), cloud service provider (*CSP*). Figure 2 shows our system model. Next we will introduce the rights of each entity one by one.

Authority. Authority, like the manager of a system, is a completely trusted entity. *AT* is responsible for generating the user's private key, system parameters and system master key. In addition, *AT* participates in user revocation. *AT* is responsible for deleting all the information of the revoked user in the system and notifying the cloud service provider to perform revocation.

Data user. *DU* has a series of attributes and attempts to access data in the system. *DU* is divided into revoked users and unrevoked users. If the attributes of the data user meet the access policy in the ciphertext and are not revoked, *DU* can decrypt the ciphertext.

Data owner. The data owner is the owner of the data transmitted in the system. *DO* has the right to determine the access policy. In order to ensure the security of the data, *DO* encrypts the data and stores it on the cloud server provider. The ciphertext contains the access structure set by *DO*.

Cloud service provider. Because the cloud service provider is not trusted, the data is stored in encrypted form in *CSP*. In addition, in our system, *CSP* is also used to implement user revocation. In the event of user revocation, *CSP* will perform revocation algorithm so that the revoked user cannot decrypt the new ciphertext and the system permissions of the unrevoked user will not be affected.

3.2 System model

As far as we know, our scheme is the first to achieve constant-size ciphertext, constant-size key, user traceability and user revocation at the same time. Therefore, different from the ordinary CP-ABE scheme, our scheme adds revocation and traceability algorithms in order to achieve high efficiency, traceability and user revocation. Next, we will specifically introduce the six algorithms included in our scheme.

Setup $(\xi, \mathcal{S}) \rightarrow (pp, msk)$**.** *AT* executes algorithm *Setup* and inputs the security parameter ξ and the universe attribute set $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ and output public parameters *pp* and master secret key *msk*.

KeyGen $(pp, id, S, msk) \rightarrow dk$**.** The algorithm is launched by *AT*. It takes public parameters *pp*, the user identity *id*, an attribute string *S*, master secret key *msk* as input, and outputs the decrypt key *dk*.

Encrypt $(\mathbb{P}, pp, m) \rightarrow ct$**.** The algorithm inputs an access policy $\mathbb{P}$, the public paremeters pp, plaintext m and outputs the ciphertext ct.

Revocation $(id, U, s_i, ct) \rightarrow ct'$**.** This algorithm is a revocation algorithm implemented by *CSP*. By updating the ciphertext, the revoked user cannot decrypt the updated ciphertext even if he or she has the corresponding key before revocation. The decryption rights of unrevoked users are not affected. The algorithm inputs user identity id, unrevoked user list U, user's secret s_i, ciphertext ct, output updated ciphertext ct'.

Decrypt $(dk, ct, pp, id, S) \rightarrow m$**.** The algorithm takes the decryption key dk, ciphertext ct, public parameters pp, user identity id, an attribute set S as input and outputs plaintext m.

Trace $(dk, pp, msk) \rightarrow (id/\top)$**.** *AT* implements this algorithm. It inputs the decrypt key dk, public parameters pp, master secret key msk and outputs the malicious user id or $\top$.

3.3 Security model

In order to prove the security of the system, we give the following selective CCA security game. The participants of the game are challenger $\mathcal{C}$ and attacker $\mathcal{A}$.

Initialization. At the beginning of the game, attacker $\mathcal{A}$ first declares the access structure $\mathbb{P}^*$ s/he wants to attack and sends it to challenger $\mathcal{C}$.

Setup. After getting a security parameter ξ, $\mathcal{C}$ executes algorithm *Setup* to obtain system parameters. Then, $\mathcal{C}$ sends public parameters to $\mathcal{A}$.

Query phase 1. At this stage, $\mathcal{A}$ sends a key query request to $\mathcal{C}$, and $\mathcal{A}$ can ask for keys other than those that satisfy the access policy $\mathbb{P}^*$. After receiving the key query, $\mathcal{C}$ executes algorithm *KenGen*. $\mathcal{C}$ sends the private key returned by algorithm *KenGen* to $\mathcal{A}$.

Challenge. After completing the key query, $\mathcal{A}$ chooses two plaintext messages of the same length, m_1, m_2. And m_1, m_2 are sent to $\mathcal{C}$ together with the access policy P^*. $\mathcal{C}$ tosses a coin at random b, $b \in \{0, 1\}$. Afterwards, $\mathcal{C}$ launched encryption algorithm with access policy $\mathbb{P}^*$. Therefore, m_b is encrypted as ciphertext ct. $\mathcal{C}$ returns the ciphertext ct to $\mathcal{A}$.

Query phase 2. The key query is performed as in **Query phase 1**. Similarly, the key queried does not include a key that satisfies the access policy $\mathbb{P}^*$.

Guess. $\mathcal{A}$ printed out his conjecture b' about b. If $b = b'$, then $\mathcal{A}$ wins this game. The probability of $\mathcal{A}$ winning the above game is defined as $Adv_{\mathcal{A},\xi} = |Pr[b' = b] - \frac{1}{2}|$.

Definition 2 If the advantages of $\mathcal{A}$ winning the above games can not be ignored in any probability polynomial time, then we say that the ABE scheme is CCA secure.

3.4 Traceability model

This paper uses the traceability model defined in [13]. The traceability security game between the attacker $\mathcal{A}$ and the challenger $\mathcal{C}$ is given below, which is described as follows.

Setup. $\mathcal{C}$ executes algorithm *Setup*. Then $\mathcal{C}$ passes the public parameter pp to $\mathcal{A}$.

Key Query. $\mathcal{A}$ asks $\mathcal{C}$ for the user's private key associated with $(id_1, S_1), \ldots, (id_q, S_q)$.

Key Forgery. $\mathcal{A}$ outputs a user private key dk^*.

If Trace $(dk^*, pp, msk) \neq \top$ (that is, dk^* is well-formed), and Trace $(dk^*, pp, msk) \notin \{id_1, \ldots, id_q\}$, where id_i is the user identity used for inquiry $(i = 1, \ldots, q)$, the attacker wins the above game. The advantage for an attacker to win the above game is defined as $\Pr[\text{Trace}(dk^*, pp, msk) \notin \{\top\} \cup \{id_1, \ldots, id_q\}]$.

Definition 3 If all polynomial time attackers have negligible advantages in the above traceability games, the scheme in this paper is traceable.

4 Construction

4.1 Setup

Get a bilinear group map $\mathcal{G} = \{e, G_1, G_2, G_T, p\}$, where h, g are the generator of G_1, G_2 severally. The algorithm chooses a probabilistic encryption scheme (*Enc*, *Dec*) from binary string to Z_p^*, and makes a_1 and a_2 as its different keys. Furthermore, the algorithm initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme, $\mathcal{INS}_{\bar{t},\bar{n}}$, and secretly stores $f(x)$ and $\bar{t} - 1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$. *AT* computes $e(h, g)$ and randomly selects $\beta_1, \beta_2, \alpha \in Z_p$. Let $w_i = h^{\alpha^i}, u_i = h^{\beta_1 \alpha^i}, v_i = h^{\beta_2 \alpha^i}, i = 1, \ldots, n$. After that, *AT* selects the following four one-way anti-collision hash functions: $H_1 : \{0,1\}^* \to \{0,1\}^{l_s}$; $H_2, H_3 : \{0,1\}^* \to Z_p^*$; $H_4 : \{0,1\}^* \to \{0,1\}^{l_m}$. Finally, the algorithm outputs the public parameters $pp = \{\mathcal{G}, H_1, H_2, H_3, H_4, e(h, g), g^\alpha, w_i, u_i, v_i\}$ and the master secret key $msk = \{g, \alpha, \beta_1, \beta_2, a_1, a_2\}$.

4.2 KeyGen

AT first calculates $x = Enc_{a_1}(id)$, $y = f(x)$, $c = Enc_{a_2}(x||y)$. Furthermore, for a user with identity *id* and attribute string $S = c_1 c_2 \ldots c_n$, *AT* calculates the following:

$$f(\alpha, S) = \prod_{i=1}^{n} (\alpha + H_3(i))^{1-c_i},$$

where $f(x, S)$ is a polynomial of order n about x.

Then, the algorithm selects a random number r_1 and calculates the value of s according to the condition: $\frac{1}{f(\alpha,S)} = \beta_1 s + \beta_2 r_1$. Therefore, $s = \frac{1}{\beta_1}\left(\frac{1}{f(\alpha,S)} - \beta_2 r_1\right)$. Finally, the decryption key is set to $dk = \{K = \frac{a_1}{\alpha+c} g^{r_1 s}, K' = c, L_1 = g^{r_1}, L_2 = g^s\}$ and output.

4.3 Encrypt

Given an access policy $\mathbb{P} = b_1 b_2 \dots b_n$, the algorithm calculates a polynomial function $f(x, \mathbb{P})$ with the highest order $n-1$ as below:

$$f(x, \mathbb{P}) = \prod_{i=1}^{n} (x + H_3(i))^{1-b_i}$$

where f_i is the coefficient of x^i. *DO* selects a random number θ from $\{0,1\}^{l_\xi}$ and calculates $r_m = H_2(\mathbb{P}, m, \theta)$ and $e(h,g)^{r_m}$. The ciphertext ct is computed as:

$$\begin{aligned}
C_0 &= H_4(\theta) \oplus m \\
C_1 &= \left(\prod_{i=0}^{n} \left(u_i\right)^{f_i}\right)^{r_m} = h^{\beta_1 f(\alpha,\mathbb{P}) r_m} \\
C_2 &= \left(\prod_{i=0}^{n} \left(v_i\right)^{f_i}\right)^{r_m} = h^{\beta_2 f(\alpha,\mathbb{P}) r_m} \\
C_3 &= (g^{\alpha})^{r_m} \\
C_4 &= H_1(e(h,g)^{r_m}) \oplus \theta
\end{aligned}$$

The algorithm outputs ciphertext $ct = \{\mathbb{P}, C_0, C_1, C_2, C_3, C_4\}$

4.4 Revocation

When the user enters the system, *CSP* randomly selects the personal secret s_i for each user (with identity information *id*), and then sends the personal secret s_i to the user through the secure channel. Let $U = \{U_1, U_2, \dots, U_t\}$ represent a collection of unrevoked users in the system. Once a user is revoked, *CSP* updates the ciphertext. *CSP* randomly selects a secret value ω, followed by two random numbers γ_1, γ_2. Next, *CSP* extracts the secret of the unrevoked user and removes the secret of the revoked user from the system. Then, *CSP* calculate the following polynomials.

$$f(t) = (\gamma_1 t - \gamma_2) A(t) + \omega$$

where $A(t) = \prod_{U_i \in U} (t - s_i)$. *CSP* calculates $C_0' = C_0 \oplus \omega = H_4(\theta) \oplus m \oplus \omega$.

Finally, *CSP* outputs the ciphertext $ct' = \{\mathbb{P}, C_0', C_1, C_2, C_3, C_4, f(t)\}$.

CSP can dynamically prevent revoked users from accessing the system. Once user revocation occurs, *CSP* updates the list of unrevoked users, recalculates authorization polynomials, and updates the ciphertext with new random secrets.

4.5 Decrypt

First, it is determined whether the user's attribute S satisfies the access policy $\mathbb{P}$ and whether the user is a unrevoked user. If any one of them is not satisfied, the algorithm is terminated. Otherwise, it calculate

$$d_i = c_i - b_i$$

$$F(x) = f(x, S, \mathbb{P}) = \prod_{i=1}^{n-|\mathbb{P}|} (x + H_3(i))^{d_i}$$

where F_i is the coefficient of x^i and $F(0) \neq 0$. Then, the algorithm computes the following formulas.

$$\begin{aligned}
X &= e\left(\prod_{i=1}^{n-|\mathbb{P}|} \left(w_{i-1}\right)^{F_i}, C_3\right) \\
&= e(h, g)^{r_m \prod_{i=1}^{n-|\mathbb{P}|} \alpha^i F_i + r_m F_0 - r_m F_0} \\
&= e(h, g)^{r_m F(\alpha) - r_m F_0} \\
Y &= e(C_1, g^s) \\
&= e(h, g)^{\beta_1 f(\alpha, \mathbb{P}) r_m s} \\
Z &= e(C_2, g^{r_1}) \\
&= e(h, g)^{\beta_2 f(\alpha, \mathbb{P}) r_m r_1} \\
\left(\frac{YZ}{X}\right)^{\frac{1}{F_0}} &= \left(\frac{e(h, g)^{r_m F(\alpha)}}{e(h, g)^{r_m F(\alpha) - r_m F_0}}\right)^{\frac{1}{F_0}} \\
&= e(h, g)^{r_m}
\end{aligned}$$

Because the following equation holds, $\omega = f(s_i)$.

$$f(s_i) = (\lambda_1 s_i - \lambda_2)(s_i - s_i) \prod_{i \neq j} (s_i - s_j) + \omega = \omega$$

$$\theta' = H_1(e(h, g)^{r_m}) \oplus C_4$$

$$m' = C_0' \oplus H_4(\theta') \oplus \omega$$

$$r_m' = H_2(\mathbb{P}, m', \theta')$$

If the equation $e(h, g)^{r_m} = e(h, g)^{r_m'}$ holds, the algorithm outputs plaintext m. Otherwise, the algorithm output $\perp$.

Correctness.

$$
\begin{aligned}
X &= e\left(\prod_{i=1}^{n-|\mathbb{P}|}(w_{i-1})^{F_i}, C_3\right) \\
&= e\left(\prod_{i=1}^{n-|\mathbb{P}|} h^{\alpha^{i-1}F_i}, g^{\alpha r_m}\right) \\
&= e(h,g)^{\alpha r_m \prod_{i=1}^{n-|\mathbb{P}|} \alpha^{i-1}F_i}) \\
&= e(h,g)^{r_m \prod_{i=1}^{n-|\mathbb{P}|} \alpha^{i}F_i + r_m F_0 - r_m F_0} \\
&= e(h,g)^{r_m F(\alpha) - r_m F_0} \\
Y &= e\left(C_1, g^s\right) \\
&= e\left(h^{\beta_1 f(\alpha,\mathbb{P}) r_m}, g^s\right) \\
&= e(h,g)^{\beta_1 f(\alpha,\mathbb{P}) r_m s} \\
Z &= e\left(C_2, g^{r_1}\right) \\
&= e\left(h^{\beta_2 f(\alpha,\mathbb{P}) r_m}, g^{r_1}\right) \\
&= e(h,g)^{\beta_2 f(\alpha,\mathbb{P}) r_m r_1} \\
YZ &= e(h,g)^{\beta_1 f(\alpha,\mathbb{P}) r_m s + \beta_2 f(\alpha,\mathbb{P}) r_m r_1} \\
&= e(h,g)^{r_m F(\alpha)} \\
\left(\frac{YZ}{X}\right)^{\frac{1}{F_0}} &= \left(\frac{e(h,g)^{r_m F(\alpha)}}{e(h,g)^{r_m F(\alpha) - r_m F_0}}\right)^{\frac{1}{F_0}} \\
&= e(h,g)^{r_m}
\end{aligned}
$$

4.6 Trace

First, the algorithm executes key sanity check. If dk_{id} doesn't passes the following check, the algorithm output is $\top$. Otherwise, it takes the next step.

(1) The constituent elements in dk_{id} are in the form of (K, K', L_1, L_2) and satisfy:$K' \in Z_p^*, K, L_1, L_2 \in G_2$.
(2) $e(K, g^{\alpha} g^{K'}) = e(L_1, L_2)^{a_1}$.

Then the algorithm extracts $(x^* = x, y^* = y)$ from $x||y = Dec(c)$ in decrypt key dk. If $(x^* = x, y^* = y) \in \{(x_1, y_1), (x_2, y_2), \ldots, (x_{t-1}, y_{t-1})\}$, the algorithm calculates $Dec_{a_1}(x^*)$ and obtains the id of the malicious user's identity. Otherwise, it proceeds to the next step.

The algorithm restores the secret ${a_0}^*$ of $\mathcal{INS}_{\bar{t},\bar{n}}$ based on $t-1$ point and $(x^* = x, y^* = y)$. If ${a_0}^* = f(0)$, the algorithm calculates the $Dec(x^*)$ and gets the malicious user id. Otherwise, it output $\top$.

5 Security analysis

In this section, we will give the security analysis of the proposed scheme. The security proof of our efficient scheme depends on the scheme [16]. In order to express conveniently and succinctly, we use **CSCK** to express the scheme [16] and **RAT-CSCK** to express our scheme that implements user revocation, user traceability, constant-size ciphertext and key.

Theorem 1 *If* **CSCK** *is selectively CCA secure, then* **RAT-CSCK** *is also selectively CCA secure.*

Proof Assuming $\mathcal{A}$ who has a non-negligible advantage to win the selectively CCA game of **RAT-CSCK**, we construct a PPT simulator algorithm $\mathcal{T}$ to break **CSCK**. $\mathcal{T}$ plays both the adversary of **CSCK** and the challenger of **RAT-CSCK**.

Initialization. $\mathcal{A}$ announces the access policy $\mathbb{P}^*$ s/he wants to attack and sends $\mathbb{P}^*$ to $\mathcal{T}$. $\mathcal{T}$ receives and sends $\mathbb{P}^*$ to challenger $\mathcal{C}$. $\mathcal{C}$ runs algorithm *Setup* and sends the output public parameters $pp = \{\mathcal{G}, H_1, H_2, H_3, H_4, e(h, g), g^\alpha, w_i, u_i, v_i\}$ and master secret key $msk = \{g, \alpha, \beta_1, \beta_2\}$ to $\mathcal{T}$.

Setup. $\mathcal{T}$ chooses a probabilistic encryption scheme (*Enc*, *Dec*) from binary string to Z_p^*, and makes a_1 and a_2 as its different keys. Furthermore, $\mathcal{T}$ initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme, $\mathcal{INS}_{\bar{t},\bar{n}}$, and secretly stores $f(x)$ and $t-1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{t-1}, y_{t-1})\}$. Then, $\mathcal{T}$ gives $pp = \{\mathcal{G}, H_1, H_2, H_3, H_4, e(h, g), g^\alpha, w_i, u_i, v_i\}$ to $\mathcal{A}$ and keeps $msk = \{g, \alpha, \beta_1, \beta_2, a_1, a_2\}$ secret.

Query Phase 1. $\mathcal{A}$ sends a query request to $\mathcal{T}$, and $\mathcal{T}$ responds as follows according to the type of query.

- Decryption key query (*id*, *S*). Once $\mathcal{T}$ receives the decryption key query request, $\mathcal{T}$ gives S to $\mathcal{C}$. $\mathcal{C}$ sends the corresponding decrypt key $dk_{\mathcal{C}} = \{g^{r_1}, g^s\}$ to $\mathcal{T}$. $\mathcal{T}$ first calculates $x = Enc_{a_1}(id)$, $y = f(x)$, $c = Enc_{a_2}(x||y)$. AT calculates the following: $f(\alpha, S) = \prod_{i=1}^{n}(\alpha + H_3(i))^{1-c_i}$, where $f(x, S)$ is a polynomial of order n about x. Then, the algorithm selects a random number r_1 and calculates the value of s according to the condition: $\frac{1}{f(\alpha,S)} = \beta_1 s + \beta_2 r_1$. Therefore, $s = \frac{1}{\beta_1}\left(\frac{1}{f(\alpha,S)} - \beta_2 r_1\right)$. For a user with identity *id* and attribute string $S = c_1 c_2 \ldots c_n$, $\mathcal{T}$ calculates the decryption key $dk = \{K = \frac{a_1}{\alpha+c} g^{r_1 s}, K' = c, L_1 = g^{r_1}, L_2 = g^s\}$. Then, $\mathcal{T}$ sends *dk* to $\mathcal{A}$.
- Decryption query (*dk*, *id*, *S*). When $\mathcal{T}$ receives the decryption request from $\mathcal{A}$, $\mathcal{T}$ checks the validity of the user. $\mathcal{T}$ finds out whether there is a corresponding *id* in the system according to the user's s_i. If the corresponding s_i does not exist, then $\mathcal{T}$ returns $\top$. Otherwise, $\mathcal{T}$ executes the decryption algorithm and sends the decrypted plaintext m to $\mathcal{A}$.

Challenge. $\mathcal{A}$ sends two adaptively selected challenge messages of equal length m_0, m_1 to $\mathcal{T}$. Afterwards, $\mathcal{T}$ submits m_0, m_1 to $\mathcal{C}$. $\mathcal{C}$ flips a coin and encrypts m_b. $\mathcal{T}$ obtains the ciphertext $CT_{\mathcal{C}} = \{\mathbb{P}, C_0, C_1, C_2, C_3, C_4\}$. $\mathcal{T}$ randomly selects a secret value ω, followed by two random numbers γ_1, γ_2. Next, $\mathcal{T}$ extracts the secret of the unrevoked user and calculates the

following polynomial: $f(t) = (\gamma_1 t - \gamma_2)A(t) + \omega$, where $A(t) = \prod_{U_i \in U}(t - s_i)$. $\mathcal{T}$ calculates $C_0' = C_0 \oplus \omega = H_4(\theta) \oplus m \oplus \omega$.

Afterwards, $\mathcal{T}$ gives the ciphertext $ct = \{\mathbb{P}, C_0', C_1, C_2, C_3, C_4, f(t)\}$.

Query Phase 2. The operation at this stage is the same as that of **Query Phase 1**, but there's two conditions: (1) when initiating a decryption key query, $\mathcal{A}$ cannot enter the attribute S that satisfies the access policy $\mathbb{P}^*$;(2) when initiating a decryption query, $\mathcal{A}$ cannot enter m^* as the challenge plaintext.

Guess. $\mathcal{A}$ sends his/her guess b' of b to $\mathcal{T}$ and $\mathcal{T}$ gives b' to $\mathcal{C}$.

6 Traceability

In this section, based on the $l - SDH$ assumption, we give the traceability proof of our scheme. We resort to the method of proof in [5] and [14].

Theorem 2 *If the $l - SDH$ assumption holds, then our scheme is traceable ($q < l$, q is the number of inquiries by the adversary).*

Proof Assuming that there is a PPT adversary $\mathcal{A}$ who can win the traceability game given in Sect. 3. D with an non-negligible advantage λ after q key queries (let $l = q + 1$), then we can construct a PPT algorithm $\mathcal{B}$ which can break the $l - SDH$ assumption with same non-negligible advantages. $\mathcal{T}$ is the challenger who interacts with the simulator $\mathcal{B}$ in this game.

Let G_1, G_2 and G_T be bilinear cyclic groups of order p, and $h, \dot{g}$ are the generator of G_1, G_2, respectively. $e : G_1 \times G_2 \to G_T$ is a bilinear mapping, $\alpha \in Z_p^*$. The following instance, $IN_{SDH} = (p, G_1, G_2, G_T, e, \dot{g}, \dot{g}^{\alpha}, \dot{g}^{\alpha^2}, \ldots, \dot{g}^{\alpha^l})$, is given. The goal of $\mathcal{B}$ is to output $\hat{c} \in Z_p$ and $r \in G_2$, and $(\hat{c}, r)$ satisfy the following equation: $r = \dot{g}^{\frac{1}{\alpha+\hat{c}}}$ so as to solve the $l - SDH$ assumption. Let $B_i = \dot{g}^{\alpha^i}, i = 0, 1, 2, \ldots, l$, then $\mathcal{B}$ inputs $(p, h, G_1, G_2, G_T, e, \{B_i\}_{i=0}^{l})$ and starts the traceability game with $\mathcal{A}$.

Setup. $\mathcal{B}$ selects q different random numbers $\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_q$ from Z_p^*. Let the polynomial $f(z) = \prod_{i=1}^{q}(z + \hat{c}_i)$. By expanding the polynomial, $\mathcal{B}$ can get an expression in the following form: $f(z) = \sum_{i=0}^{q} \theta_i z^i$. θ_i is the corresponding coefficient of the polynomial expansion. $\mathcal{B}$ calculates g and g^{α}.

$$g = \prod_{i=0}^{q}(B_i)^{\theta_i} = \prod_{i=0}^{q}(\dot{g})^{\theta_i \alpha^i} = \dot{g}^{f(\alpha)}$$

$$g^{\alpha} = \prod_{i=1}^{q+1}(B_i)^{\theta_{i-1}} = \prod_{i=0}^{q}(\dot{g})^{\theta_i \alpha^i \alpha} = \dot{g}^{f(\alpha)\alpha}$$

$\mathcal{B}$ randomly selects $\beta_1, \beta_2 \in Z_p$. Let $w_i = h^{\alpha^i}, u_i = h^{\beta_1 \alpha^i}, v_i = h^{\beta_2 \alpha^i}, i = 1, \ldots, n$. After that, the algorithm selects the following four one-way anti-collision hash functions: $H_1 : \{0,1\}^* \to \{0,1\}^{l_\xi}$; $H_2, H_3 : \{0,1\}^* \to Z_p^*$; $H_4 : \{0,1\}^* \to \{0,1\}^{l_m}$. $\mathcal{B}$ sends the public parameter $pp = \{\mathcal{G}, H_1, H_2, H_3, H_4, e(h,g), g^\alpha, w_i, u_i, v_i\}$ to $\mathcal{A}$.

Key Query. $\mathcal{A}$ submits (id_i, S_i) to $\mathcal{B}$ and queries the associated user private key dk. Suppose this is the i-th $(i \le q)$ inquiry of $\mathcal{A}$.

Let the polynomial $f_i(z) = \frac{f(z)}{z+\hat{c}_i} = \prod_{j=1, j\neq i}^{q} (z + \hat{c}_j)$. Expand the polynomial and get an expression in the following form $f_i(z) = \sum_{j=0}^{q-1} \rho_j z^j$. ρ_j is the corresponding coefficient of the polynomial expansion. $\mathcal{B}$ calculates

$$\delta_i = \prod_{j=0}^{q-1} (B_j)^{\rho_j} = \prod_{j=0}^{q-1} (\dot{g})^{\rho_j \alpha^j} = \dot{g}^{\frac{f(\alpha)}{\alpha+\hat{c}_i}} = g^{\frac{1}{\alpha+\hat{c}_i}}$$

Then, $\mathcal{B}$ selects a random number r_1 and calculates the value of s according to the condition: $\frac{1}{f(\alpha,S)} = \beta_1 s + \beta_2 r_1$. Therefore, $s = \frac{1}{\beta_1}\left(\frac{1}{f(\alpha,S)} - \beta_2 r_1\right)$. Next, $\mathcal{B}$ calculates the user private key component:

$$K = (\delta_i)^{a_1} g^{r_1 s} = g^{\frac{a_1}{\alpha+\hat{c}_i}} g^{r_1 s}, K' = \hat{c}_i, L_1 = g^{r_1}, L_2 = g^s$$

$\mathcal{B}$ passes the user's private key $dk_i = \{K, K', L_1, L_2\}$ to $\mathcal{A}$. dk_i represents the user's private key obtained by the i-th query of $\mathcal{A}$.

Key Forgery. $\mathcal{A}$ submits the user's private key dk^* to $\mathcal{B}$. It is worth noting that the public parameters and user private keys simulated by $\mathcal{B}$ in the traceability game are the same as the distribution in the actual scheme.

Let $P_{\mathcal{A}}$ means $\mathcal{A}$ wins the traceability game,i.e. dk^* satisfies the key sanity check and K' in dk^* does not belong to $\{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_q\}$. From the assumption at the beginning of the proof, there is $\Pr[P_{\mathcal{A}}] = \lambda$. When the event $P_{\mathcal{A}}$ occurs, $\mathcal{B}$ calculates this formula $\frac{f(z)}{z+K'}$. The quotient is $\Delta(z) = \sum_{i=0}^{q-1} \eta_i z^i$ and the remainder is $\overline{\eta} \in Z_p$. $\overline{\eta} \neq 0$, since $f(z) = \prod_{i=1}^{q} (z + \hat{c}_i), \hat{c}_1, \ldots, \hat{c}_q \in Z_p^*$ and $K' \notin \{\hat{c}_1, \ldots, \hat{c}_q\}$. That is, $f(z)$ doesn't been divided by $(z + K')$. Then $\mathcal{A}$ write $f(z)$ as $f(z) = \Delta(z)(z + K') + \overline{\eta}$. Because p is a prime, $\overline{\eta} \in Z_p$ and $\overline{\eta} \neq 0$, $\overline{\eta}$ and p are mutually prime, i.e. their greatest common divisor $gcd(\overline{\eta}, p) = 1$. Therefore, $\overline{\eta}$ has an inverse element $\frac{1}{\overline{\eta}} mod p$ under module p. At this point, $\mathcal{B}$ can calculate $(\hat{c}, r)$ as follows.

According to the key format check condition, let's assume $L_2 = g^s$, then we have $K = g^{\frac{a_1}{\alpha+K'}} g^{r_1 s}$. $\mathcal{B}$ computes $\frac{1}{\overline{\eta}} mod p$ and

$$\delta = \left(\frac{K}{{L_2}^{r_1}}\right)^{{a_1}^{-1} \ mod \ p} = \left(\frac{g^{\frac{a_1}{\alpha+K'}} g^{r_1 s}}{(g^s)^{r_1}}\right)^{{a_1}^{-1} \ mod \ p}$$

$$= g^{\frac{1}{\alpha+K'}} = \dot{g}^{\frac{f(\alpha)}{\alpha+K'}} = \dot{g}^{\frac{\Delta(\alpha)(\alpha+K')+\overline{\eta}}{\alpha+K'}} = \dot{g}^{\Delta(\alpha)} \dot{g}^{\frac{\overline{\eta}}{\alpha+K'}}$$

$$r = \left(\delta \prod_{i=0}^{q-1} \left(B_i\right)^{-\eta_i}\right)^{\frac{1}{\bar{\eta}}} = \left(\dot{g}^{\Delta(\alpha)} \dot{g}^{\frac{\bar{\eta}}{\alpha+K'}} \prod_{i=0}^{q-1} \dot{g}^{-\eta_i \alpha^i}\right)^{\frac{1}{\bar{\eta}}}$$
$$= \left(\dot{g}^{\Delta(\alpha)} \dot{g}^{\frac{\bar{\eta}}{\alpha+K'}} \dot{g}^{-\Delta(\alpha)}\right)^{\frac{1}{\bar{\eta}}} = \dot{g}^{\frac{1}{\alpha+K'}}$$

$\hat{c} = K' \bmod p$

Because $e\left(\dot{g}^{\alpha}\dot{g}^{\hat{c}}, r\right) = e\left(\dot{g}^{\alpha}\dot{g}^{K'}, \dot{g}^{\frac{1}{\alpha+K'}}\right) = e(\dot{g}, \dot{g})$, $(\hat{c}, r)$ is a solution of the $l - SDH$ assumption, there is an equation $\Pr = [P_{SDH}(\hat{c}, r)|P_{\mathcal{A}}] = 1$, where $P_{SDH}(\hat{c}, r)$ means $(\hat{c}, r)$ is a solution of the $l - SDH$ assumption.

To sum up, the probability of $\mathcal{B}$ breaking the $l - SDH$ assumption is

$$\begin{aligned} Pr\left[P_{SDH}(\hat{c}, r)\right] &= Pr\left[P_{SDH}(\hat{c}, r)|\overline{P_{\mathcal{A}}}\right] Pr\left[\overline{P_{\mathcal{A}}}\right] \\ &\quad + Pr\left[P_{SDH}(\hat{c}, r)|\overline{P_{\mathcal{A}}}\right] Pr\left[\overline{P_{\mathcal{A}}}\right] \\ &= 0 \times (1 - \lambda) + 1 \times \lambda = \lambda \end{aligned}$$

Among them, the probability of $\mathcal{B}$ solving the $l - SDH$ assumption without any help is considered to be negligible, and for the convenience of calculation, it is set to 0. Then the advantage of $\mathcal{B}$ in breaking the $l - SDH$ assumption is

$$Adv_{\mathcal{B},SDH} = |Pr[P_{SDH}(\hat{c}, r)] - 0| = |\lambda - 0| = \lambda$$

Therefore, $\mathcal{B}$ has the advantage that λ can not be ignored to break the $l - SDH$ assumption.

7 Comparison

In this part, our scheme will be compared with other related schemes [21, 20, 16] in terms of performance. The content of comparison mainly involves two aspects, theoretical analysis and experimental comparison. Among them, the theoretical analysis is mainly through the artificial calculation of the ciphertext length, the key length and the operation time of encryption and decryption. Because the calculation time of exponentiation operation and pairing operation is much longer than that of other operations, we only consider exponentiation operation and pairing operation. As for the experimental comparison, we will give the encryption time complexity and decryption time complexity of each scheme through experimental simulation.

7.1 Theoretical analysis

In Table 1, we compare the functional features of different schemes. Scheme [21] implements traceable users, but it does not achieve constant-size ciphertext and key. This scheme takes a lot of computing time and storage space. Meanwhile, for the users with malicious behavior, the revocation is not realized, which affects the confidentiality of the scheme. Although scheme [20] implements user revocation, it is time-consuming, equal to $\left((5l + 4 + 2I)E_G + P + E_{G_T}\right)$, where I represents the number of all i such that $t[i] = 0$, and l represents the number of attributes in the access policy. The cost of our scheme for user revocation is $2O$, that is, the cost of two XOR operations. It can be seen

Table 1 Function Comparison

Scheme	Traceable user	Constant-size ciphertext	Constant-size key	Storage for tracing	User revocation
Yan et al. [21]	✓	✗	✗	Linear	✗
Xu et al. [20]	✗	✗	✗	✗	✓$((5l+4+2I)E_G+P+E_{G_T})$
Odelu et al. [16]	✗	✓	✓	✗	✗
Our	✓	✓	Constant	✓	✓(2O)

that the cost of user revocation in our scheme is much lower than that in scheme [20]. In addition, scheme [20] does not take into account the problem of key abuse, in addition, the scheme does not achieve constant-size ciphertext and constant-size key. Scheme [16], like scheme [20], does not consider the problem of malicious users abusing keys. In addition, the problem of user rights change in ABE system has not been solved. Our scheme realizes constant-size ciphertext and constant-size key, which not only ensures the efficiency of the scheme, but also solves the problem of key abuse and user permission change in the system, which can not only trace malicious users, but also revoke the system permissions of malicious users in a timely manner. By comparison, it is found that our scheme is the most comprehensive and functional.

The comparative result of theoretical analysis is shown in Table 2. Because cloud servers are considered to have strong storage and computing capacity, we only consider local computing in computing. L_G, L_{G_T} and L_{Z_p} denote the length of an element on G, G_T and Z_p, respectively. Use E_G to denote the time it takes to perform a exponential operation on G. Use E_{G_T} to denote the time of an exponential operation on G_T. P is used to denote the time consumed by a pairing operation. D represents the length of the time code in the scheme. l represents the number of attributes in the access policy. n is used to represent the number of attributes in the global attribute set. We can see from Table 2 that the key length and ciphertext length of our scheme are much smaller than those of scheme [21, 20]. And they do not increase with the increase in the number of global attributes and the number of attributes in the access policy. In terms of the time consumed by encryption and decryption, our scheme and scheme [16] are the same, and both are lower than schemes [21] and [20]. This shows that our scheme is efficient enough and requires less storage space and transmission time.

Table 2 Efficiency Comparison

Scheme	Key length	Ciphertext length	Encryption	Decryption
Yan et al. [21]	$(3+n)L_G+2L_{Z_p}$	$(2+3l)L_G+L_{G_T}$	$(2+5l)E_G+E_{G_T}$	$(1+3l)P+(1+l)E_G+lE_{G_T}$
Xu et al. [20]	$(2+n)L_G$	$(2+3l+D)L_G+L_{G_T}$	$(2+5l+D)E_G+E_{G_T}$	$(2+3l)P+lE_{G_T}$
Odelu et al. [16]	$2L_G$	$3L_G+2L_{Z_p}$	$(2n+3)E_G+P+E_{G_T}$	$(n-l+2)P+(n-l)E_G+E_{G_T}$
Our	$2L_G+L_{Z_p}$	$3L_G+2L_{Z_p}$	$(2n+3)E_G+P+E_{G_T}$	$(n-l+2)P+(n-l)E_G+E_{G_T}$

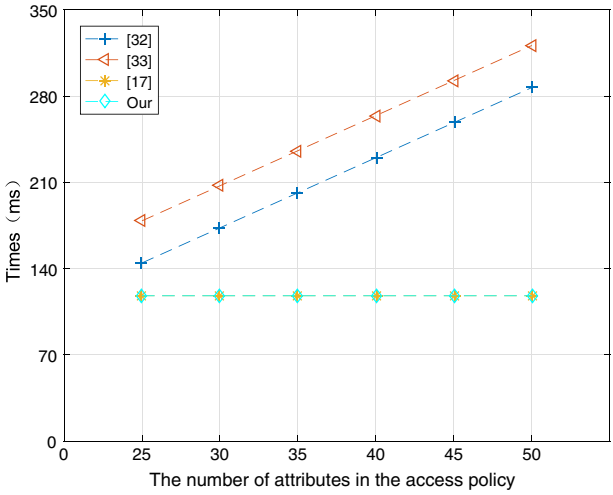

Fig. 3 Encryption complexity

7.2 Experimental comparison

We have carried out simulation experiments on our scheme and these schemes [21, 20, 16]. We use a symmetrical elliptic curve "SS512" with a base field of 512 bits to implement these four schemes. These experiments are run on a Windows 10 system with AMD8400U and 16GB RAM, which is configured with Matlab and PBC library. The experimental results are shown in Figs. 3 and 4.

Generally speaking, our scheme has obvious advantages over [21, 20] in terms of encryption complexity and decryption complexity. The complexity of encryption and decryption is similar to that of scheme [16], but our scheme is more functional than scheme [16]. In addition, we can see from Figs. 3 and 4 that the encryption complexity of our scheme do not increase with the increase of the number of attributes. It is a straight line approximately parallel to the axis of the number of attributes. At the same time, the

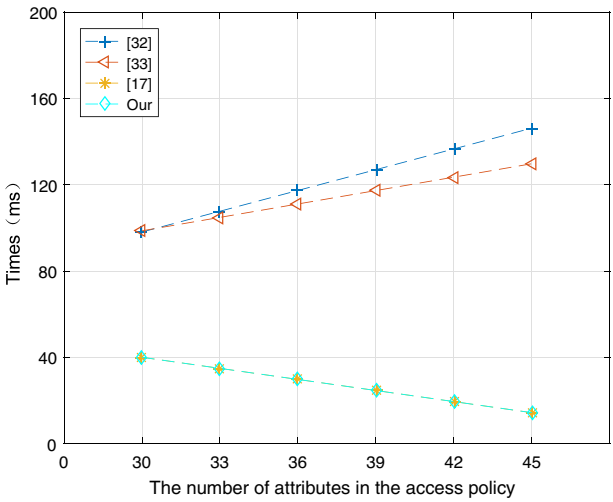

Fig. 4 Decryption complexity

decryption complexity of our scheme decreases with the increase of attributes. This is exactly what we hope to achieve an efficient ABE system.

8 Conclusion and future work

In this work, we propose a traceable and revocable attribute-based encryption system with constant-size ciphertext and key, which can be used in IoT. Specifically, we implement the traceability of misbehavior users in the system. During the traceability process, the storage overhead of the system used for traceability is constant and will not increase with the increase of users in the system. In addition, it implements the revocation of misbehavior users and users who log out of the system. By updating the ciphertext, even if the revoked user gets the updated ciphertext, s/he cannot decrypt the ciphertext with her/his own private key. Moreover, in order to make the system more efficient, we introduce the characteristics of constant-size ciphertext and key, so that the ciphertext length and key length are independent of the corresponding number of attributes, which significantly shortens the ciphertext, key transmission time and user decryption time, and improves the efficiency of the scheme. Compared with the existing schemes, we can see that our scheme has obvious advantages in terms of functional features and efficiency. Finally, we prove that our scheme is selectively CCA secure under the standard model.

In this work, we implement white-box traceability for misbehavior users. In our future work, we will focus on a stronger concept of black-box traceability. In black-box traceability, misbehavior users disclose the decryption device rather than the decryption key. Specifically, the misbehavior users could hide the decryption key and decryption algorithm by tweaking it. In this case, because the decryption key and decryption algorithm are not well-formed, our proposed traceable and revocable attribute-based encryption scheme with constant-size ciphertext and key will fail. It will be necessary to construct a black-box traceable and revocable attribute-based encryption scheme with a constant-size ciphertext and key. In addition, the flexible access structure is the embodiment of the performance of a CP-ABE scheme. In the following work, we will try our best to achieve a more flexible access structure to improve the expression ability of the system.

References

1. Attrapadung N, Imai H (2009) Attribute-based encryption supporting direct/indirect revocation modes. In Cryptography & Coding, Ima International Conference, Cryptography & Coding. Cirencester, UK
2. Attrapadung N, Libert B, De Panafieu E (2011) Expressive key-policy attribute-based encryption with constant-size ciphertexts. pp 90–108
3. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In IEEE Symposium on Security & Privacy

4. Boneh D, Gentry C, Waters B (2005) Collusion resistant broadcast encryption with short ciphertexts and private keys. pp 258–275
5. Dan B, Boyen X (2004) Short signatures without random oracles
6. Doshi N, Jinwala DC (2014) Fully secure ciphertext policy attribute-based encryption with constant length ciphertext and faster decryption. Secur Commun Netw 7(11):1988–2002
7. Emura K, Miyaji A, Nomura A, Omote K, Soshi M (2009) A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. pp 13–23
8. 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
9. Green M, Hohenberger S, Waters B (2011) Outsourcing the decryption of ABE ciphertexts. pp 34–34
10. Guo F, Mu Y, Susilo W, Wong DS, Varadharajan V (2014) Cp-abe with constant-size keys for lightweight devices. IEEE Trans Inf Forensics Secur 9(5):763–771
11. Li J, Huang Q, Chen X, Chow SSM, Wong DS, Xie D (2011) Multi-authority ciphertext-policy attribute-based encryption with accountability. pp 386–390
12. Li J, Ren K, Kim K (2009) A2be: Accountable attribute-based encryption for abuse free access control". IACR Cryptology ePrint Archive 2009:118
13. Liu Z, Cao Z, Wong DS (2013) White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. IEEE Trans Inf Forensics Secur 8(1):76–88
14. Ning J, Dong X, Cao Z, Wei L, Lin X (2015) White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. IEEE Trans Inf Forensics Secur 10(6):1274–1288
15. Odelu V, Das AK, Khan MK, Choo KR, Jo M (2017) Expressive cp-abe scheme for mobile devices in iot satisfying constant-size keys and ciphertexts. IEEE Access 5:3273–3283
16. Odelu V, Das AK, Rao YS, Kumari S, Khan MK, Choo KKR (2017) Pairing-based CP-ABE with constant-size ciphertexts and secret keys for cloud environment. Comput Stand Interfaces 54(PT.1):3–9
17. Peng Z, Chen Z, Liang K, Wang S, Wang T (2016) A cloud-based access control scheme with user revocation and attribute update
18. Sahai A, Waters B (2005) Fuzzy identity-based encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 457–473
19. Xu Z, Martin KM (2012) Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage. pp 844–849
20. Xu S, Yang G, Mu Y, Liu X (2019) A secure IoT cloud storage system with fine-grained access control and decryption key exposure resistance. Futur Gener Comput Syst 97:284–294
21. Yan X, He X, Yu J, Tang Y (2019) White-box traceable ciphertext-policy attribute-based encryption in multi-domain environment. IEEE Access 7:128298–128312
22. Zhang Y, Chen X, Li J, Li H, Li F (2013) FDR-ABE: Attribute-based encryption with flexible and direct revocation. pp 38–45
23. Zhang Y, Li J, Yan H (2019) Constant size ciphertext distributed CP-ABE scheme with privacy protection and fully hiding access structure. IEEE Access 7:47982–47990
24. Zhen L, Cao Z, Wong DS (2017) Traceable CP-ABE: How to trace decryption devices found in the wild. IEEE Trans Inf Forensics Secur 10(1):55–68

Authors and Affiliations

Yi Wu[1] · Wei Zhang[1] · Hu Xiong[1] · Zhiguang Qin[1] · Kuo-Hui Yeh[2]

Yi Wu
201821090101@std.uestc.edu.cn

Hu Xiong
xionghu.uestc@gmail.com

Zhiguang Qin
qinzg@uestc.edu.cn

Kuo-Hui Yeh
khyeh@gms.ndhu.edu.tw

[1] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China

[2] Department of Information Management, National Dong Hwa University, 97401 Hualien, Taiwan, ROC