The ISC Int'l Journal of Information Security

January 2023, Volume 15, Number 1 (pp. 97–110)

http://www.isecure-journal.org

Highly Efficient and Revocable CP-ABE with Outsourcing Decryption for IoT **

Sina Abdollahi [1,*], Javad Mohajeri [2], and Mahmoud Salmasizadeh [2]

[1] *Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran.*

[2] *Electronics Research Institute, Sharif University of Technology, Tehran, Iran.*

ARTICLE INFO.

Article history:

Received: December 24, 2021

Revised: June 9, 2022

Accepted: August 24, 2022

Published Online: August 27, 2022

Keywords:

Attribute-Based Encryption, Internet of Things, Outsourced Decryption, Revocation, Constant Number of Pairings

Type: Research Article

doi: 10.22042/isecure.2022.321360.738

dor: 20.1001.1.20082045.2023.15.1.4.1

ABSTRACT

Ciphertext-policy attribute-based encryption(CP-ABE) is considered a promising solution for secure data sharing in the cloud environment. Although very well expressiveness in ABE constructions can be achieved using a linear secret sharing scheme(LSSS), there is a significant drawback in such constructions. In the LSSS-based ABE construction, the number of heavy pairing operations increases with an increase in the number of required attributes in the decryption. In this paper, we propose an LSSS-based CP-ABE scheme with a fixed number of pairings(four pairings) during the decryption process. In our scheme increasing the number of required attributes in the decryption does not affect the number of pairings. The simulation shows that our scheme has significant advantages in the encryption and the decryption processes compared to previous schemes. In addition, we use the outsourcing method in the decryption to get better performance on the user side. The main burden of decryption computation is done by the cloud without revealing any information about the plaintext. Furthermore, in our revocation method, the users' communication channels are not used during the revocation process. All of these features make our scheme suitable for applications such as IoT. The proposed scheme is selectively CPA-secure in the standard model.

1 Introduction

Cloud computing provides a variety of cost-effective features for users [1]. Cloud capabilities for encrypted data storage have become an interesting topic in the last few years. Among the types of access control methods [2] which can be used in such a system, attribute-based access control (ABAC) is proposed as a suitable and efficient method. ABAC is a logical method that grants or denies access to the system resources, not through a pre-defined user ID but, through the "attributes" of the subject and object. Each subject and object are related to several attributes and specific relations between the subject's attributes and the object's attributes will lead to the subject's access to the object. In this kind of access control, adding new subjects to the system does not change the object's access roles. It is a very important feature, especially in a dynamic environment like IoT systems. Attribute-based encryption (ABE) is a public key encryption method that provides confidentiality and ABAC simultaneously. In an ABE

* Corresponding author.

**This article is an extended/revised version of an ISCISC'18 paper.

Email addresses: sina.abdollahi@ee.sharif.edu, mohajer@sharif.edu, salmasi@sharif.edu

ISSN: 2008-2045

system, each data user(subject) receives a secret key according to its attributes. For example, being a student is an attribute that applies to all students in the system. Furthermore, each plaintext(object) is encrypted under several attributes. If there is a special relationship between the user's attributes and the plaintext's attributes, decryption will be possible otherwise, the user obtain no information about the plaintext. Attribute-based encryption is a "one to many" public key encryption method. This means that the data users' identities are not involved in the encryption process and every user with an appropriate set of attributes can decrypt the ciphertext.

Secret sharing schemes [3] are mainly used in ABE constructions. Each secret-sharing scheme creates an access structure that is satisfied by certain combinations of attributes. Therefore, ABE schemes are divided into two main categories [4]. In one category, each ciphertext is associated with a set of attributes, and each user's secret key is associated with an access structure. Such a method is called key-policy attribute-based encryption (KP-ABE). In another category, each ciphertext is associated with an access structure and each user's secret key is associated with a set of attributes. This method is called ciphertext-policy attribute-based encryption (CP-ABE). The ciphertext-policy ABE allows the encryptor to get complete information about the attributes set of users who can decrypt the ciphertext. To date, the second method has received more attention.

The lack of efficiency limits the application of ABE schemes. For instance, e-health and IoT systems consist of limited energy storage devices, and it is infeasible for these devices to perform heavy computations such as a large number of pairing. Furthermore, sending lots of messages on communication channels is not possible. These actions consume a lot of energy and so, to design an ABE scheme for an application like IoT, computational and communication costs have to be at least as possible. In this paper, we have addressed these two issues.

The rest of the paper is organized as follows: in the continuation of this section, our motivation for doing this work and the main ideas of our work are expressed. Also, related works are introduced. In Section 2 we briefly introduce the preliminaries of our paper and in Section 3 we present a basic construction for a CP-ABE scheme. In Section 4 the proposed scheme is introduced and in Section 5 security features of our scheme will be proved. Section 6 consists of performance evaluation and the paper ends with the conclusion in Section 7.

1.1 Our Motivation and Contribution

Pairing operations are mainly used in ABE schemes and create a large computational overhead in the decryption process. Especially, in more expressive access structures, an increase in the number of attributes required for the decryption leads to an increase in the number of pairings. This creates a huge load in the decryption process. In recent years, several papers have proposed CP-ABE schemes using elliptic curve cryptography which are very efficient due to the lack of pairing but these schemes have their drawbacks. Reference [5] suggests a method to break these schemes, which according to our studies can be used to break several elliptic curve-based CP-ABE schemes including [6–8]. On the other hand, reaching a straightforward security proof is easier in pairing-based cryptography. Therefore, we use pairing-based cryptography and at the same time try to use elliptic curve cryptography methods to reach better efficiency. For example, similar to elliptic curve cryptography-based schemes, we use field elements in public parameters, ciphertexts, and users' keys. Consequently, a large part of the decryption is done by field multiplication and scalar multiplication without the need for pairing. In the proposed scheme, independent of the number of required attributes in the decryption, only four pairings are used in the decryption. The proposed scheme asymptotically has equal computational efficiency to the mentioned elliptic curve-based schemes and is secure under a well-studied pairing-based computational assumption.

Although we eliminate the relationship between the number of pairings and the number of required attributes in the decryption, still there exist several scalar multiplications with a considerable load. Therefore, to get better performance on the user side, the main decryption computation is outsourced. Without revealing any information about the plaintext, the main decryption computation is performed by the cloud which is called "partial decryption". As a result, each user completes the decryption with only one exponentiation.

In a real ABE system, users' attributes change frequently. In addition, each attribute is shared among several users. So, any single-attribute revocation of a user may affect other users. Therefore, designing an efficient method for attribute revocation is a challenging and important problem. On the other hand, using the outsourced decryption arises the idea of revocation in an outsourced manner. We use a user-independent revocation method, which is rarely seen in previous schemes. In our revocation method, for each data user, the cloud receives an outsourced key from the trusted authority which is necessary for par-

tial decryption. By setting these keys, the trusted authority controls the ability or inability of the cloud to perform partial decryption. Consequently, the revocation has been implemented independently of users' secret keys.

Specifically, our contribution includes the following:

- We propose a highly efficient CP-ABE scheme with a completely flexible access structure. In the proposed scheme only four pairings are used during each decryption process. The simulations show the high efficiency of our scheme in encryption and decryption.
- In our revocation method, the users' communication channels are not used during the revocation process. Therefore, compared to other revocation methods, there is a significant improvement in the communication cost on the user side.
- In the proposed scheme, the main parts of the decryption are outsourced to the cloud. The cloud executes the main part of the decryption without obtaining any information about the plaintext.
- The proposed scheme is selectively CPA-secure under the Decisional q-Bilinear Diffie-Hellman exponent assumption in the standard model.

1.2 Related Work

Attribute-based encryption was first proposed in [9] and as the basic ABE schemes, we mention to KP-ABE schemes [9, 10] and CP-ABE schemes [4, 11, 12]. Outsourcing the main decryption computation to the third party was first proposed in [13] and in addition to the outsourcing decryption, other issues have been addressed in [14–20]. Reference [14] presents a fully distributed structure on the trusted authorities to get better performance in scalability and flexibility, references [15, 16] present verifiability methods in the outsourced decryption to the third party, in [17–19] in addition to the outsourced decryption, outsourcing the main encryption computation to other entities are introduced, and [20] introduce the white box traceability of the malicious users and a method for changing access policy in the existing ciphertexts.

Recently, some papers have replaced pairing with lighter operations. For instance, in [21] an RSA-based scheme is introduced but it has a limitation on the access structure expressiveness. In [22, 23] elliptic curve-based KP-ABE schemes are introduced. Reference [24] is an elliptic curve-based CP-ABE scheme but in this scheme, the trusted authority is used in the decryption process. Actively using the trusted authority in the decryption process, puts a lot of computational burden on the trusted authority which is inappropriate according to the traditional model of ABE schemes. Several other elliptic curve-based CP-ABE schemes are introduced in [6–8, 24]. Unfortunately, the security proof of these schemes is incorrect. Reference [5] introduces an attack to [6, 24] which according to our studies can be used to break [6–8]. This attack uses field elements in an elliptic curve-based scheme to form a system of linear equations which leads to revealing secret parameters.

A wide range of revocation methods has been used in ABE schemes. The revocation method in [25] is "direct". In the direct revocation method, the revocation list is embedded directly into ciphertexts. In this method, all users map into a binary tree, and non-revoked users are uniquely presented by some nodes of the tree. Related information of these nodes is embedded into all ciphertexts in a way that only non-revoked users with enough attributes for satisfying the access structures, can decrypt the ciphertexts. When a user is revoked, nodes' information changes, consequently, all ciphertexts must be changed. Although the revocation process does not change the users' secret keys, embedding related information of the revoked users leads to a considerable increase in the size of the ciphertext.

In [26, 27] "server aided" method is used for revocation. In this method, the key generation center(KGC) periodically sends key updates to an entity which is called an untrusted server. Moreover, the binary tree is used to reduce the size of key updates from linear to logarithmic order. The untrusted server executes the main decryption computation (partial decryption) without revealing any information about the plaintext and the user completes the decryption by its secret key. The key updates determine the untrusted server's ability to execute partial decryption for each user in that period. Although users' secret keys are fixed and never change during different periods, user revocation occurs at the beginning of certain periods and immediate revocation is not possible.

The revocation method in [14] is "indirect". It means that to revoke a user's attribute, secret keys of other users (users holding this attribute) and all ciphertexts which use this attribute (in the access structure), must be changed. In this paper, immediate revocation has been implemented, but changing the secret key of users during each revocation process creates a huge communication load on the user side.

2 Preliminaries

2.1 Bilinear Map

Let $\mathbb{G}$ be an additive cyclic group of prime order p with generator P and $\mathbb{G}_T$ as a multiplicative cyclic group of the same order. e is a bilinear map

$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ if it has the following properties:
1) Bilinearity: $\forall U, V \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(aU, bV) = e(U, V)^{ab}$
2) Non-degeneracy: $e(P, P) \neq 1$
3) Computability: $e(U, V)$ is computable in a polynomial-time algorithm for any $U, V \in \mathbb{G}$.

Like some other sources, in the rest of the paper, we call e pairing.

2.2 Complexity Assumption

We use the decisional q-bilinear Diffie-Hellman exponent(q-BDHE) as the computational assumption similar to [12].

Definition 1 (q-BDHE). At first, two random elements $a, b \in \mathbb{Z}_p$, a random point $R \in \mathbb{G}_T$, a random bit δ and the vector $\overrightarrow{Q} = (P, aP, a^2P, \ldots, a^qP, a^{q+2}P, a^{q+3}P, \ldots, a^{2q}P, bP)$ are generated. If $\delta = 0$, we have $T = e(P, P)^{a^{q+1}b}$, otherwise $T = R$. Then the tuple $(\overrightarrow{Q}, T)$ is given to the adversary. If the decisional q-BDHE exponent assumption is held, for every PPT [1] algorithm $\mathcal{B}$, there is a negligible value ε such that:

$$|Pr[\mathcal{B}(\overrightarrow{Q}, e(P,P)^{a^{q+1}b}) = 0] - Pr[\mathcal{B}(\overrightarrow{Q}, R) = 0]| \leq \varepsilon \quad (1)$$

2.3 Access Structure and LSSS

Definition 2 (Access Structure). Consider a set of parties $W = \{P_1, P_2, \ldots, P_n\}$. An access structure is a collection $\mathbb{A} \subseteq 2^W$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets. A collection $\mathbb{A}$ is a monotone access structure if $\forall B, C : B \in \mathbb{A}$ and $B \subseteq C$ implies that $C \in \mathbb{A}$.

In an ABE scheme, monotone access structures are commonly used. Any monotone access structure can be implemented by a linear secret sharing scheme (LSSS) [3]. So LSSS-based access structure is considered the most expressive access structure.

Definition 3 (LSSS). A secret sharing scheme Π over a set of parties W, is called linear if:
1- The shares for each party form a vector over $\mathbb{Z}$.
2- There exists a share generating matrix M with t rows and n columns and the function ρ that maps each row of the matrix M to an attribute. The column random vector $\overrightarrow{v} = (s, v_1, v_2, \ldots, v_{n-1}) \in \mathbb{Z}_p^n$ is generated and s is the secret to be shared. Each row x of vector $M\overrightarrow{v}$ belongs to the party $\rho(x)$.

Based on linear reconstruction property [3], for each authorized set $S \in \mathbb{A}$ and $\mu = \{x : \rho(x) \in S\}$, there are constants $\{c_x\}_{x \in \mu}$ such that if $\{\lambda_x\}_{x \in \mu}$ are valid shares, then s can be reconstructed by the relation $\sum_{x \in \mu} c_x \lambda_x = s$. It is shown that these constants $\{c_x\}$ can be found in a polynomial-time algorithm [3].

For each unauthorized set $S' \notin \mathbb{A}$ and a defined set $\mu' = \{x : \rho(x) \in S'\}$, there exist a vector $\overrightarrow{\omega} = (\omega_1, \omega_2, \ldots, \omega_n)$ such that: $\omega_1 = 1, \forall x \in \mu' : M_x . \overrightarrow{\omega} = 0$

[1] Probabilistic Polynomial-Time

3 Basic Ciphertext-Policy ABE

3.1 System Model and Definitions

In this section, we briefly define a basic structure for the CP-ABE schemes based on [12]. Each CP-ABE scheme consists of three main entities trusted authority (TA), data owners (DOs), and data users (DUs) and four algorithms **Setup**, **KeyGen**, **Encryption**, and **Decryption**. The TA is responsible for generating system parameters by execution of **Setup** algorithm and users' key generation by executing **KeyGen** algorithm. Ciphertexts are generated by the execution of **Encryption** algorithm. Each ciphertext is associated with an access structure (M, ρ) which defines the access roles to the plaintext. A data user with an appropriate set of attributes can decrypt a ciphertext by execution of the **Decryption** algorithm. These algorithms are defined as follows.

- **Setup**$(\lambda) \to (PP, MSK)$: The algorithm input is the security parameter λ and its outputs are public parameters PP, and master secret key MSK.
- **KeyGen**$(MSK, S_{du}) \to sk_{du}$: The algorithm inputs are the master secret key MSK and DU's attribute set S_{du}. The algorithm output is DU's secret key sk_{du}.
- **Encryption**$(PP, m, (M, \rho)) \to ct$: The algorithm inputs are the public parameters PP, a plaintext m, and an access structure (M, ρ). The algorithm output is ciphertext ct.
- **Decryption**$(PP, ct, sk_{du}) \to m$ *or* $\perp$: The inputs of the algorithm are the public parameters PP, the ciphertext ct, and the DU's secret key sk_{du}. If the attributes in the DU's secret key satisfy the ciphertext access structure, the output is the plaintext m, and otherwise, the output is equal to $\perp$.

3.2 Security Model

As we have already mentioned, in an ABE scheme decryption will be possible if the data user's attributes satisfy the ciphertext access structure. Therefore, a secure ABE scheme has to guarantee that several colluding users with insufficient sets of attributes can not decrypt the ciphertext. To ensure resistance against this kind of attack, traditionally an indistinguishability game is defined as follows.

- **Initialization.** The adversary $\mathcal{A}$ sends the challenge access structure (M^*, ρ^*) to the simulator $\mathcal{S}$.
- **Setup.** $\mathcal{S}$ runs the **Setup** algorithm to generate public parameters and give them to $\mathcal{A}$.
- **Phase1.** $\mathcal{S}$ responds to $\mathcal{A}$ queries for key generation. The adversary $\mathcal{A}$ makes queries on different sets of attributes with the restriction that each set can not satisfy the challenge access structure (M^*, ρ^*). $\mathcal{S}$ provides the corresponding secret key sk_{du} to $\mathcal{A}$ in an execution of **KeyGen** algorithm.
- **Challenge.** $\mathcal{A}$ sends two plaintexts m_0, m_1 to $\mathcal{S}$. Then $\mathcal{S}$ selects a random bit γ and executes the **Encryption** $(PP, m_\gamma, (M^*, \rho^*)) \rightarrow ct_\gamma$ algorithm. Finally, challenge ciphertext ct_γ is sent to $\mathcal{A}$.
- **Phase2.** It is the same as **Phase1**.
- **Guess.** The adversary generates bit γ' as a guess of γ and sends it to $\mathcal{S}$. The advantage of the adversary in this game is equal to:

$$Adv_{\mathcal{A}}(\lambda) = |Pr[\gamma = \gamma'] - \frac{1}{2}|$$

Definition 4. A CP-ABE scheme (with the four mentioned algorithms in Section 3.1) is selectively CPA-secure if any PPT adversary has at most a negligible advantage in the above game.

The adversary selects the challenge access structure (M^*, ρ^*) before the setup phase. According to the challenge, access structure attributes are divided into two groups. One group is the attributes in the challenge access structure and the other group consists of other attributes in the universe. The simulator acts differently in the generation of public parameters for each group. This strategy is called partitioning which is often used in the security proof of ABE schemes.

4 Proposed Scheme

4.1 System Model

We add the cloud to the basic model of Section 3.1 as an entity with high computing and storage capabilities. Therefore, our model consists of four entities: trusted authority (TA), the cloud, data owners(DOs), and data users(DUs). The system model is shown in Figure 1. The trusted authority is responsible for DU's key generation. Each DU requests for secret key on a specific set of attributes. After authenticating, the TA sends DU's secret key sk_{du} to the DU and the DU's outsourced key ok_{du} to the cloud. The outsourced key is a part of DU's key which frequently changes during the revocation processes. In addition, the security proof will be shown that these outsourced keys do not affect the security of the system (see our security model in Section 4.3). A DO uses the public parameters PP for the encryption and sends the generated ciphertext to the cloud. Therefore, there are two databases in the cloud which are related to the outsourced keys and the ciphertexts. These databases are shown in Figure 1. We assume that these databases are publicly visible to all entities. The decryption also involves several steps. At first, a DU announces a ciphertext to the cloud. The cloud performs partial decryption using the outsourced key ok_{du} and the transform ciphertext tct is generated. Finally, the DU uses the transform ciphertext tct and the secret key sk_{du} to complete the decryption.

For revoking an attribute of a DU, several steps must be done. These steps are shown in Figure 2. For revoking an attribute I of a DU, the outsourced key of other users who hold attribute I and the ciphertexts which use this attribute in their access structure must be changed. At first, the TA generates a re-encryption key RK_I and new public parameters PP'. The cloud uses the re-encryption key RK_I to generate a re-encrypted version for the related ciphertexts. Moreover, the TA generates new DUs' outsourced attribute keys $ok'_{I,du}$ for all non-revoked DUs which are shown by the set A in Figure 2. The cloud replaces the older version of these keys with new ones and the new outsourced keys ok'_{du} are obtained.

4.2 Definitions

In the proposed scheme, revocation is added to the four algorithms of the basic structure in Section 3.1. Therefore, the proposed scheme consists of five main algorithms: Setup, Key Generation, Encryption, Decryption, and Revocation. Each main algorithm consists of one or more algorithms which are described as follows.

Setup. This phase consists of one algorithm. This algorithm is executed by the TA and generates the necessary public and secret parameters of the system.

- **Setup**$(\lambda) \rightarrow (PP, MSK)$: The algorithm input is the security parameter λ and its outputs are public parameters PP, and master secret key MSK.

Key Generation. This phase consists of one algorithm. The **KeyGen** algorithm is executed by the TA for each DU in the system and generates user-related keys. The TA sends the DU's secret key sk_{du} to the DU and the DU's outsourced key ok_{du} to the cloud.

- **KeyGen**$(MSK, S_{du}) \rightarrow (sk_{du}, ok_{du})$: The algorithm inputs are the master secret key MSK and the DU's attribute set S_{du}. The outputs of the algorithm are the DU's secret key sk_{du} and the DU's outsourced key ok_{du}.

Encryption. This phase consists of one algorithm. The

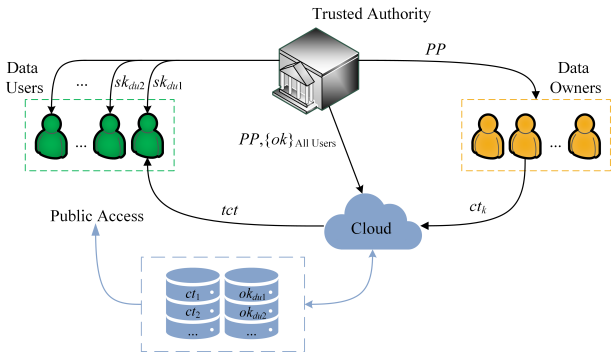


Figure 1. System model

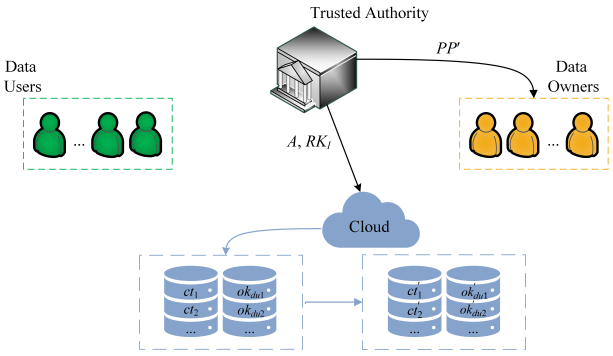


Figure 2. Revocation process $A=\{ok'_{I,du}\}_{\text{For attribute I owners}}$

DO executes the **Encryption** algorithm to generate the ciphertext.

- **Encryption**$(PP, m, (M, \rho)) \rightarrow ct$: The algorithm inputs are the public parameters PP, a plaintext m, and an access structure (M, ρ). The algorithm output is the ciphertext ct.

Decryption. We divide the decryption process into two algorithms which are **Partial-Decryption** and **Decryption**. The main overhead of the decryption computations is performed in the **Partial-Decryption** algorithm. As the outsourced keys are publicly available, the DUs can perform **Partial-Decryption** by themselves but it is obvious that doing this task by the cloud is preferable. Therefore, we consider that this algorithm is executed by the cloud. Finally, the DU completes the decryption process by executing the **Decryption** algorithm.

- **Partial-Decryption**$(PP, ok_{du}, ct) \rightarrow tct$ *or* $\perp$: The inputs of the algorithm are the public parameters PP, the DU's outsourced key ok_{du}, and the ciphertext ct. If the attributes in the DU's outsourced key satisfy the ciphertext access structure, the output is the transform ciphertext tct, and otherwise, the output is equal to $\perp$.
- **Decryption**$(sk_{du}, tct) \rightarrow m$: The inputs of the algorithm are the DU's secret key sk_{du} and the transform ciphertext tct. The algorithm output is the plaintext m.

Revocation. This phase consists of three algorithms **Update-PP**, **Update-Key**, and **Re-Encryption**. The DUs' channels are not used during the revocation process. The TA in collaboration with the cloud executes these algorithms in such a way that the revoked DU's outsourced attribute key becomes useless. The first two algorithms are executed by the TA and the last is executed by the cloud. The **Update-PP** generates necessary updated public parameters, the **Update-Key** generates new user-related parameters which are executed for DUs and **Re-Encryption** generates a re-encrypted version for a ciphertext.

- **Update-PP**$(PP,MSK,I)\rightarrow(PP',MSK',RK_I)$: The algorithm inputs are the public parameters PP, the master secret key MSK, and an attribute I. The algorithm outputs are new public parameters PP', new master secret key MSK', and the re-encryption key RK_I.
- **Update-Key**$(PP', MSK', I, ok_{du}) \rightarrow ok'_{I,du}$: The algorithm inputs are the new public parameters PP', the new master secret key MSK', the attribute I, and the DU's outsourced key ok_{du}. The output is the new DU's outsourced attribute keys $ok'_{I,du}$.
- **Re-Encryption**$(PP', I, RK_I, ct) \rightarrow ct'$: The algorithm inputs are the public parameters PP, attribute I, the re-encryption key RK_I, and the ciphertext ct. The algorithm output is the re-encrypted ciphertext ct'.

4.3 Security Model

For each user, the cloud receives an outsourced key which is publicly visible to all entities. Therefore, in addition to the adversary's abilities in the basic security game (Section 3.2), another kind of query should be considered. In the defined security game, first-type queries represent the adversary's ability to collect several data user-related keys and second-type queries represent the adversary's ability to collect the outsourced keys of all DUs in the system. The security game is defined as follows.

- **Initialization.** The adversary $\mathcal{A}$ sends the challenge access structure (M^*, ρ^*) to the simulator $\mathcal{S}$.
- **Setup.** $\mathcal{S}$ executes the **Setup** algorithm to generate public parameters which are given to $\mathcal{A}$.
- **Phase1.** $\mathcal{S}$ responds to $\mathcal{A}$ queries for key generation. $\mathcal{A}$ queries include the following:
  1) First-type query: $\mathcal{A}$ makes queries on different sets of attributes with the restriction that

these sets can not satisfy (M^*, ρ^*). $\mathcal{S}$ provides the corresponding keys $\{sk_{du}, ok_{du}\}$ to $\mathcal{A}$ by executing **KeyGen** algorithm.
2) Second-type query: $\mathcal{A}$ makes queries on different sets of attributes without any restriction and $\mathcal{S}$ provides the corresponding outsourced key ok_{du} to $\mathcal{A}$ by executing **KeyGen** algorithm.

- **Challenge.** $\mathcal{A}$ sends two plaintexts m_0, m_1 to $\mathcal{S}$. The simulator $\mathcal{S}$ selects a random bit γ and executes the **Encryption** $(PP, m_\gamma, (M^*, \rho^*)) \rightarrow ct_\gamma$ algorithm. Finally, the challenge ciphertext ct_γ is sent to $\mathcal{A}$.
- **Phase2.** It is the same as **Phase1**.
- **Guess.** The adversary generates bit γ' as a guess of γ and sends it to $\mathcal{S}$. The advantage of $\mathcal{A}$ in this game is equal to:

$$Adv_{\mathcal{A}}(\lambda) = |Pr[\gamma = \gamma'] - \frac{1}{2}|$$

Definition 5. Our CP-ABE scheme is selectively CPA-secure in the standard model if any PPT adversary has at most a negligible advantage in the above game.

4.4 Construction

Setup$(\lambda) \rightarrow (PP, MSK)$: At first, the groups $\mathbb{G}$ and $\mathbb{G}_T$ are generated based on λ. The attributes universe is equal to $U = \mathbb{Z}_p$. Then random elements $\alpha, \beta \in \mathbb{Z}_p$ and also, for each attribute $i \in U$, random elements $u_i, k_i \in \mathbb{Z}_p$ are generated. The algorithm outputs are generated as follows.

$$pk_{1,i} = k_i + u_i(modp) \quad pk_{2,i} = u_i \beta P$$

$$PP = \{P, \beta P, e(P,P)^{\alpha\beta}, \{pk_{1,i}, pk_{2,i}\}_{i \in U}\}$$

$$MSK = \{\alpha, \beta, \{k_i\}_{i \in U}\}$$

KeyGen$(MSK, S_{du}) \rightarrow (sk_{du}, ok_{du})$: At first, the random elements $z, r \in \mathbb{Z}_p$ and a unique identity ID_{du} are generated. For each attribute $i \in S_{du}$, a random element $r_i \in \mathbb{Z}_p$ is generated and the outputs are equal to:

$$K_{1,i} = (\alpha + r)k_i + r_i(modp) \quad K_{2,i} = \beta r_i P$$

$$ok_{i,du} = \{K_{1,i}, K_{2,i}\}$$

$$K_1 = zr\beta P \quad K_2 = (\alpha + r)P$$

$$sk_{du} = \{ID_{du}, S_{du}, z\}$$

$$ok_{du} = \{ID_{du}, S_{du}, K_1, K_2, \{ok_{i,du}\}_{i \in S_{du}}\}$$

Encryption$(PP, m, (M, \rho)) \rightarrow ct$: The size of matrix M is equal to $t \times n$ and the plaintext is $m \in \mathbb{G}$. The algorithm generates a random element $w \in \mathbb{Z}_p$ and a random column vector $\overrightarrow{v} = (s, v_1, \ldots, v_{n-1})^T \in \mathbb{Z}_p^n$. Furthermore, we define λ_x as $\lambda_x = (M\overrightarrow{v})_x$. For each attribute in the access structure, random element $w_{\rho(x)} \in \mathbb{Z}_p$ is generated and the ciphertext ct is equal to:

$$C = me(P,P)^{\alpha\beta s} \quad C_1 = wP \quad C_2 = sP$$

$$C_{1,x} = wpk_{2,\rho(x)} + w_{\rho(x)}\beta P$$

$$C_{2,x} = wpk_{1,\rho(x)} + w_{\rho(x)} - \lambda_x(modp)$$

$$ct = \{(M, \rho), C, C_1, C_2, \{C_{1,x}, C_{2,x}\}_{x=[1:t]}\}$$

ρ is an injective function. It means that each attribute i is related to at most one row of the matrix M. In other words, an attribute cannot be used twice in the share generating matrix [2].

Partial-Decryption$(PP, ok_{du}, ct) \rightarrow tct$ *or* $\perp$: This algorithm defines a set $\mu = \{x : \rho(x) \in S\}$. If this set does not satisfy the ciphertext access structure, the output of the algorithm will be equal to $\perp$. Otherwise, due to the linear reconstruction property(Section 2.3), the constant $\{c_x\}_{x \in \mu}$ are found in such a way that $\sum_{x \in \mu} c_x \lambda_x = s$. Using the outsourced key ok_{du}, the transform ciphertext tct is generated as follows.

$$d_1 = \sum\nolimits_{x \in \mu} c_x K_{1,\rho(x)}(modp)$$

$$d_2 = \sum\nolimits_{x \in \mu} c_x C_{2,x}(modp)$$

$$d = d_1 C_1 - d_2 K_2 \quad d' = \sum\nolimits_{x \in \mu} c_x K_{2,\rho(x)}$$

$$d'' = \sum\nolimits_{x \in \mu} c_x C_{1,x}$$

$$tct_1 = \frac{Ce(d', C_1)}{e(d, \beta P)e(d'', K_2)} \tag{2}$$

$$tct_2 = e(C_2, K_1)$$

$$tct = \{tct_1, tct_2\}$$

Decryption$(sk_{du}, tct) \rightarrow m$: The DU computes the plaintext as follows.

$$m = tct_1 (tct_2)^{\frac{1}{z}} \tag{3}$$

Update-PP$(PP, MSK, I) \rightarrow (PP', MSK', RK_I)$: The algorithm generates random elements $k'_I, u'_I \in \mathbb{Z}_p$ and the outputs are equal to:

$$MSK' = \{\alpha, \beta, k'_I, \{k_i\}_{i \in U - I}\}$$

$$pk'_{1,I} = k'_I + u'_I(modp) \quad pk'_{2,I} = u'_I \beta P$$

$$PP' = \{\beta P, e(P,P)^{\alpha\beta}, \{pk'_{1,I}, pk'_{2,I}\}, \{pk_{1,i}, pk_{2,i}\}_{i \in U - I}\}$$

$$RK_I = (k_I - k'_I)\beta(modp)$$

Update-Key$(PP', MSK', I, ok_{du}) \rightarrow ok'_{I,du}$: At first, a random element $r'_I \in \mathbb{Z}_p$ is generated, and

[2] It is a necessary limitation for our security proof.

the new outsourced attribute key $ok'_{I,du}$ is sent to the cloud.

$$K'_{1,I} = r'_I(modp) \quad K'_{2,I} = r'_I\beta P - k'_I\beta(K_2)$$

$$ok'_{I,du} = \{K'_{1,I}, K'_{2,I}\}$$

Re-Encryption$(PP', I, RK_I, ct) \rightarrow ct'$: At first, random element $w'_I \in \mathbb{Z}_p$ is generated and ct' is generated as follows.

$$C'_{1,\rho^{-1}(I)} = C_{1,\rho^{-1}(I)} + w'_I\beta P + RK_I(wP)$$

$$C'_{2,\rho^{-1}(I)} = C_{2,\rho^{-1}(I)} + w'_I$$

$ct' = \{(M,\rho), C, C_1, C_2, \{C'_{1,x}, C'_{2,x}\}_{x=\rho^{-1}(I)}, \{C_{1,x}, C_{2,x}\}_{x=[1:t],\rho(x)\neq I}\}$

4.5 Correctness

It must be proved that a user can decrypt a ciphertext if its set of attributes satisfies the ciphertext access structure. In other words, equation (3) is correct.

Proof. At first, we obtain d_1, d_2, d, d', and d'' as follows.

$$d_1 = \sum_{x\in\mu} c_x K_{1,\rho(x)}(modp) = \sum_{x\in\mu}(\alpha + r)c_x k_{\rho(x)} + \sum_{x\in\mu} c_x r_{\rho(x)}(modp) \quad (4)$$

$$d_2 = \sum_{x\in\mu} c_x C_{2,x}(modp) = \sum_{x\in\mu} c_x(wpk_{1,\rho(x)} + w_{\rho(x)} - \lambda_x)(modp) = \sum_{x\in\mu} c_x(wk_{\rho(x)} + wu_{\rho(x)} + w_{\rho(x)}) - s(modp) \quad (5)$$

d is obtained using (4) and (5).

$$d = d_1C_1 - d_2K_2 = (\sum_{x\in\mu}(\alpha + r)c_x k_{\rho(x)} + \sum_{x\in\mu} c_x r_{\rho(x)})wP - (\sum_{x\in\mu} c_x(wk_{\rho(x)} + wu_{\rho(x)} + w_{\rho(x)}) - s)(\alpha + r)P = (w\sum_{x\in\mu} c_x r_{\rho(x)} - (\alpha + r)(w\sum_{x\in\mu} c_x u_{\rho(x)} + \sum_{x\in\mu} c_x w_{\rho(x)} - s))P \quad (6)$$

$$d' = \sum_{x\in\mu} c_x K_{2,\rho(x)} = \beta\sum_{x\in\mu} c_x r_{\rho(x)}P \quad (7)$$

$$d'' = \sum_{x\in\mu} c_x C_{1,x} = \sum_{x\in\mu} c_x(wpk_{2,\rho(x)} + w_{\rho(x)}\beta P) = \sum_{x\in\mu} c_x(wu_{\rho(x)}\beta P + w_{\rho(x)}\beta P) = (w\beta\sum_{x\in\mu} c_x u_{\rho(x)} + \beta\sum_{x\in\mu} c_x w_{\rho(x)})P \quad (8)$$

Then A, A', and A'' are defined as follows.

$$e(d, \beta P) = e(P,P)^A \quad (9)$$

$$e(d', C_1) = e(P,P)^{A'} \quad (10)$$

$$e(d'', K_2) = e(P,P)^{A''} \quad (11)$$

From (6), (7) and (8) it follows that:

$$A = \beta w\sum_{x\in\mu} c_x r_{\rho(x)} - \beta(\alpha + r)(w\sum_{x\in\mu} c_x u_{\rho(x)} + \sum_{x\in\mu} c_x w_{\rho(x)} - s)$$

$$A' = \beta w\sum_{x\in\mu} c_x r_{\rho(x)}$$

$$A'' = (\alpha + r)(w\beta\sum_{x\in\mu} c_x u_{\rho(x)} + \beta\sum_{x\in\mu} c_x w_{\rho(x)})$$

And so:

$$A' - A - A'' = -\beta(\alpha + r)s = -\beta\alpha s - \beta rs \quad (12)$$

Based on (2), (9), (10), (11) , and (12) we have:

$$tct_1 = \frac{Ce(d', C_1)}{e(d, \beta P)e(d'', K_2)} = \frac{Ce(P,P)^{A'}}{e(P,P)^A e(P,P)^{A''}} = me(P,P)^{\alpha\beta s}e(P,P)^{A'-A-A''} = me(P,P)^{-\beta rs} \quad (13)$$

And also:

$$tct_2 = e(C_2, K_1) = e(P,P)^{\beta szr} \quad (14)$$

Based on (13) and (14) it is concluded that:

$$tct_1(tct_2)^{\frac{1}{z}} = me(P,P)^{-\beta sr}e(P,P)^{(\beta szr)(\frac{1}{z})} = m$$

Therefore equation (3) is correct. □

5 Security Proof

Theorem 1. *The proposed scheme is secure (based on Definition 5) if the decisional q-BDHE assumption is held and the number of columns in the challenge access structure n^* satisfies $n^* \leq q$.*

Proof. We create a simulator $\mathcal{S}$ that uses the adversary's advantage to solve the decisional q-BDHE problem. Then, it is proved that if the decisional q-BDHE assumption is held, any PPT adversary has at most a negligible advantage in our defined game. At first, the challenger generates a random bit δ and sends q-BDHE tuple $(\overrightarrow{Q}, T)$ to the $\mathcal{S}$. The simulator $\mathcal{S}$ executes the security game(Section 4.3) as follows.

Initialization. $\mathcal{A}$ sends the challenge access structure (M^*, ρ^*) to $\mathcal{S}$. The size of the matrix M^* is equal to $t^* \times n^*$ and $n^* \leq$ q.

Setup. In this step, $\mathcal{S}$ provides the public parameters $PP = \{P, \beta P, e(P,P)^{\alpha\beta}, \{pk_{1,i}, pk_{2,i}\}_{i\in U}\}$ to $\mathcal{A}$. At first, $\mathcal{S}$ generates random elements $\alpha', \beta' \in \mathbb{Z}_p$ and defines $\alpha = \alpha' + a^q$ and $\beta = \beta' a$. So:

$$\beta P = \beta' aP = \beta'(aP)$$

$$e(P,P)^{\alpha\beta} = e(P,P)^{(\alpha'+a^q)(\beta' a)} = e(P,P)^{\alpha'\beta' a}e(P,P)^{a^{q+1}\beta'} = e(aP,P)^{\alpha'\beta'}e(a^qP, aP)^{\beta'}$$

For each attribute $i \in U, \mathcal{S}$ generates the random elements $k'_i, u'_i \in \mathbb{Z}_p$. Then the set R is defined as $R =$

$\{\rho^*(x)\}_{x=[1:t^*]}$. If $i \in R$, there is a unique x such that $\rho^*(x) = i$. x is unique due to the limitations of the share generating matrix(see **Encryption** algorithm in Section 4.4). So $\mathcal{S}$ defines k_i as follows.

$$k_i = k'_i/a + M^*_{x,1} + M^*_{x,2}a + \cdots + M^*_{x,n^*}a^{n^*-1} = k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1}$$

If $i \notin R$, $\mathcal{S}$ defines $k_i = k'_i/a$.
For each attribute i, $\mathcal{S}$ defines $u_i = -k_i + u'_i$ and so:

$$pk_{1,i} = k_i + u_i = k_i - k_i + u'_i = u'_i \tag{15}$$

If $i \in R$, $\mathcal{S}$ generates $pk_{1,i}$ as follows.

$$pk_{2,i} = u_i\beta P = (-k_i + u'_i)(\beta' a)P = -(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1})(\beta' a)P + u'_i(\beta' a)P = -k'_i\beta' P - \beta' \sum_{j=1}^{n^*} M^*_{x,j}(a^j P) + u'_i\beta'(aP) \tag{16}$$

If $i \notin R$, $\mathcal{S}$ defines $pk_{2,i}$ as follows.

$$pk_{2,i} = u_i\beta P = (-k_i + u'_i)(\beta' a)P = -k'_i\beta' P + u'_i\beta'(aP)$$

The distribution of the defined parameters is the same as the **Setup** algorithm. Consequently, our **Setup** phase is correct. It is also clear that according to the definitions and relations, $\mathcal{S}$ can generate all the elements of the public parameters PP. These two features are observed in all steps of the simulation and will not be mentioned again.

Phase1. $\mathcal{S}$ responds to $\mathcal{A}$ queries as follows.

1) First-type query: For responding to each query (on the attributes set S_{du}), $\mathcal{S}$ finds a vector $\overrightarrow{\omega} = (\omega_1, \omega_2, \ldots, \omega_{n^*})$ such that $\omega_1 = 1$ and for all x where $\rho^*(x) \in S_{du} : \overrightarrow{\omega}.M^*_x = 0$. Then $\mathcal{S}$ generates random elements $r', z \in \mathbb{Z}_p$ and also a unique identity ID_{du}. Then r is defined as follows:

$$r = r' + \omega_2 a^{q-1} + \omega_3 a^{q-2} + \cdots + \omega_{n^*}a^{q-n^*+1} = r' + \sum_{y=2}^{n^*} \omega_y a^{q-y+1}$$

Consequently, $\mathcal{S}$ generates K_1 and K_2 as follows.

$$K_1 = zr\beta P = z(r' + \sum_{y=2}^{n^*} \omega_y a^{q-y+1})\beta' aP = zr'\beta'(aP) + z\beta' \sum_{y=2}^{n^*} \omega_y(a^{q-y+2}P)$$

$$K_2 = (\alpha + r)P = \alpha' P + (a^q P) + (r' + \sum_{y=2}^{n^*} \omega_y a^{q-y+1})P = \alpha' P + (a^q P) + r'P + \sum_{y=2}^{n^*} \omega_y(a^{q-y+1}P)$$

For every attribute $i \in S_{du}$, simulator generates a random element $r'_i \in \mathbb{Z}_p$ and defines $r_i = -(\alpha+r)k_i + r'_i$. So:

$$K_{1,i} = (\alpha + r)k_i + r_i = r'_i$$

Before generating $K_{2,i}$ we obtain $(\alpha+r)k_i$ in a simpler form. If $i \in R$, we have:

$$(\alpha + r)k_i = (\alpha' + a^q + r' + \sum_{y=2}^{n^*} \omega_y a^{q-y+1})k_i = (\alpha' + r' + \sum_{y=1}^{n^*} \omega_y a^{q-y+1})k_i = (\alpha' + r' + \sum_{y=1}^{n^*} \omega_y a^{q-y+1})(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1}) = (\alpha' + r')(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1}) + k'_i\sum_{y=1}^{n^*}\omega_y a^{q-y} + \sum_{y=1}^{n^*}\sum_{j=1}^{n^*}\omega_y M^*_{x,j}a^{q-y+j} \tag{17}$$

Furthermore, we know that $\overrightarrow{\omega}.M^*_x = 0$. So:

$$\sum_{y=1}^{n^*}\sum_{j=1}^{n^*}\omega_y M^*_{x,j}a^{q-y+j} = \sum_{y=1}^{n^*}\sum_{j=1,j\neq y}^{n^*}\omega_y M^*_{x,j}a^{q-y+j} \tag{18}$$

From (17) and (18) it follows that for $i \in R$:

$$(\alpha + r)k_i = (\alpha' + r')(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1}) + k'_i\sum_{y=1}^{n^*}\omega_y a^{q-y} + \sum_{y=1}^{n^*}\sum_{j=1,j\neq y}^{n^*}\omega_y M^*_{x,j}a^{q-y+j} \tag{19}$$

If $i \notin R$, $(\alpha + r)k_i$ is equal to:

$$(\alpha + r)k_i = (\alpha' + a^q + r' + \sum_{y=2}^{n^*} \omega_y a^{q-y+1})(k'_i/a) = (\alpha' + r' + \sum_{y=1}^{n^*} \omega_y a^{q-y+1})(k'_i/a) \tag{20}$$

So, for $i \in R$, $K_{2,i}$ is generated using (19) as follows.

$$K_{2,i} = \beta r_i P = \beta' a(-(\alpha + r)k_i + r'_i)P = -(\alpha + r)k_i\beta' aP + r'_i\beta' aP = -((\beta' a)(\alpha' + r')(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j}a^{j-1}) + (\beta' a)k'_i\sum_{y=1}^{n^*}\omega_y a^{q-y} + (\beta' a)\sum_{y=1}^{n^*}\sum_{j=1,j\neq y}^{n^*}\omega_y M^*_{x,j}a^{q-y+j})P + r'_i\beta'(aP) = -\beta'(\alpha' + r')k'_i P - \beta'(\alpha' + r')\sum_{j=1}^{n^*} M^*_{x,j}(a^j P) - \beta' k'_i\sum_{y=1}^{n^*}\omega_y(a^{q-y+1}P) - \beta'\sum_{y=1}^{n^*}\sum_{j=1,j\neq y}^{n^*}\omega_y M^*_{x,j}(a^{q-y+j+1}P) + r'_i\beta'(aP)$$

For $i \notin R$, $K_{2,i}$ is generated using (20):

$$K_{2,i} = \beta r_i P = \beta' a(-(\alpha + r)k_i + r'_i)P = -(\alpha + r)k_i\beta' aP + r'_i\beta' aP = -(\alpha' + r' + \sum_{y=1}^{n^*} \omega_y a^{q-y+1})k'_i\beta' P + r'_i\beta' aP = -(\alpha' + r')k'_i\beta' P - k'_i\beta'\sum_{y=1}^{n^*}\omega_y(a^{q-y+1}P) + r'_i\beta'(aP)$$

2) Second-type query: For responding to each query (on the attributes set S'_{du}) $\mathcal{S}$ generates $z', r' \in \mathbb{Z}_p$ randomly and also a unique identity ID_{du}. Then, defines $r = r' - a^q$ and $z = z'/a$. Consequently, K_1 and K_2 are generated as follows.

$$K_1 = zr\beta P = (z'/a)(r' - a^q)\beta' aP =$$
$$z'r'\beta' P - z'\beta'(a^q P)$$
$$K_2 = (\alpha + r)P = (\alpha' + a^q + r' - a^q)P = (\alpha' + r')P$$

For every attribute $i \in S'$ the simulator generates $r'_i \in \mathbb{Z}_p$ randomly and defines $r_i = -(\alpha + r)k_i + r'_i$. So:

$$K_{1,i} = (\alpha + r)k_i + r_i = r'_i$$

If $i \in R$, the simulator generates $K_{2,i}$ as follows.

$$K_{2,i} = \beta r_i P = \beta' a(-(\alpha' + a^q + r' - a^q)k_i + r'_i)P =$$
$$-\beta' a(\alpha' + r')(k'_i/a + \sum_{j=1}^{n^*} M^*_{x,j} a^{j-1})P + \beta' a r'_i P =$$
$$-\beta'(\alpha' + r')k'_i P - \beta'(\alpha' + r')\sum_{j=1}^{n^*} M^*_{x,j}(a^j P)$$
$$+\beta' r'_i(aP)$$

If $i \notin R$, the simulator generates $K_{2,i}$ as follows.

$$K_{2,i} = \beta r_i P = \beta' a(-(\alpha + r)k_i + r'_i)P =$$
$$-\beta' a(\alpha' + a^q + r' - a^q)(k'_i/a)P + \beta' a r'_i P =$$
$$-\beta'(\alpha' + r')k'_i P + \beta' r'_i aP =$$
$$-\beta'(\alpha' + r')k'_i P + \beta' r'_i(aP)$$

Challenge. The adversary $\mathcal{A}$ sends plaintexts m_0, m_1 to $\mathcal{S}$. The simulator $\mathcal{S}$ generates $w' \in \mathbb{Z}_p$ and bit γ randomly. $\mathcal{S}$ defines $s = b$ and $w = w' + b$, so C, C_1 and C_2 are generated as follows.

$$C = m_\gamma e(P,P)^{\alpha\beta s} = m_\gamma e(P,P)^{(\alpha' + a^q)(\beta' a)b} =$$
$$m_\gamma e(P,P)^{\alpha'\beta' ab} e(P,P)^{a^{q+1}\beta' b} =$$
$$m_\gamma e(aP, bP)^{\alpha'\beta'} T^{\beta'}$$
$$C_1 = wP = (w' + b)P = w'P + (bP)$$
$$C_2 = sP = (bP)$$

$\mathcal{S}$ generates $v'_2, v'_3, ..., v'_{n^*} \in \mathbb{Z}_p$ randomly. Then, $\mathcal{S}$ generates random element $w'_{\rho(x)} \in \mathbb{Z}_p$ for each $i \in R$ and defines $w_{\rho(x)}$ and $\overrightarrow{v}$ as follows.

$$w_{\rho(x)} = b\sum_{j=1}^{n^*} M^*_{x,j} a^{j-1} - bu'_{\rho(x)} + w'_{\rho(x)} \quad (21)$$
$$\overrightarrow{v} = (v_1, v_2, ..., v_{n^*}) =$$
$$(b, ba + v'_2, ba^2 + v'_3, ..., ba^{n^*-1} + v'_{n^*})^T$$

So:

$$\lambda_x = (M^* \overrightarrow{v})_x = \sum_{j=1}^{n^*} M_{x,j} v_j =$$
$$\sum_{j=1}^{n^*} M^*_{x,j} ba^{j-1} + \sum_{j=2}^{n^*} M^*_{x,j} v'_j \quad (22)$$

Using (16) and (21), $C_{1,x}$ is generated as follows.

$$C_{1,x} = wpk_{2,\rho(x)} + w_{\rho(x)}\beta P =$$
$$(w' + b)(-k'_{\rho(x)}\beta' - \beta' \sum_{j=1}^{n^*} M^*_{x,j} a^j + u'_{\rho(x)}\beta' a)P +$$
$$(b\sum_{j=1}^{n^*} M^*_{x,j} a^{j-1} - bu'_{\rho(x)} + w'_{\rho(x)})(\beta' a)P =$$
$$-w'(k'_{\rho(x)}\beta' + \beta' \sum_{j=1}^{n^*} M^*_{x,j} a^j - u'_{\rho(x)}\beta' a)P$$
$$- k'_{\rho(x)}\beta'(bP) + w'_{\rho(x)}\beta'(aP) =$$
$$- w'k'_{\rho(x)}\beta' P - w'\beta' \sum_{j=1}^{n^*} M^*_{x,j}(a^j P) +$$
$$w'u'_{\rho(x)}\beta'(aP) - k'_{\rho(x)}\beta'(bP) + w'_{\rho(x)}\beta'(aP)$$

Using (15), (21) and (22), $C_{2,x}$ is generated as follows.

$$C_{2,x} = wpk_{1,\rho(x)} + w_{\rho(x)} - \lambda_x = (w' + b)u'_{\rho(x)} +$$
$$b\sum_{j=1}^{n^*} M^*_{x,j} a^{j-1} - bu'_{\rho(x)} + w'_{\rho(x)} - \sum_{j=1}^{n^*} M^*_{x,j} ba^{j-1} -$$
$$\sum_{j=2}^{n^*} M^*_{x,j} v'_j = w'u'_{\rho(x)} + w'_{\rho(x)} - \sum_{j=2}^{n^*} M^*_{x,j} v'_j$$

Phase2. It is the same as **Phase1**.

Guess. $\mathcal{A}$ outputs a gauss γ' of γ. If $\mathcal{A}$ guesses correctly, $\mathcal{S}$ output 0 and otherwise, the output is 1. So:

$$Pr[\mathcal{S}(\overrightarrow{Q}, e(P,P)^{a^{q+1}b}) = 0] =$$
$$Pr[\gamma = \gamma' | \delta = 0] \quad (23)$$
$$Pr[\mathcal{S}(\overrightarrow{Q}, R) = 0] = Pr[\gamma = \gamma' | \delta = 1] \quad (24)$$

From (1), (23), (24) it follows that:

$$|Pr[\gamma = \gamma' | \delta = 0] - Pr[\gamma = \gamma' | \delta = 1]| \leq \varepsilon \quad (25)$$

Based on q-BDHE complexity assumption(Section 2.2), if $\delta = 0$, T is equal to $e(P,P)^{a^{q+1}b}$ and the game is equal to the defined security game(Section 4.3) from the adversary's point of view.

$$Pr[\gamma = \gamma' | \delta = 0] =$$
$$Pr[\mathcal{A} \text{ Success in the Proposed Scheme}] \quad (26)$$

Moreover, if $\delta = 1$, T is equal to a random element R. In this case, m_γ is encrypted by the random element R and the game is equal to the indistinguishability of the OTP(One-Time Pad) cryptosystem from the adversary's point of view. So:

$$Pr[\gamma = \gamma' | \delta = 1] =$$
$$Pr[\mathcal{A} \text{ Success in OTP}] = \frac{1}{2} \quad (27)$$

Combining (25), (26) and (27) results that:

$$|Pr[\mathcal{A} \text{ Success in The Proposed Scheme}] - \frac{1}{2}| \leq \varepsilon$$
$$\Rightarrow Adv_{\mathcal{A}}(\lambda) \leq \varepsilon$$

□

Table 1. Functionality compariso. F_1: Outsourced Decryption, F_2: No Need for Token Generation in the Outsourced Decryption, F_3: Revocation, F_4: Immediate Revocation, F_5: Revocation Method Without Increasing Length of Ciphertexts, F_6: Revocation Process Independent of Data Users

Scheme	Access Structure	Decryption		Revocation			
		F_1	F_2	F_3	F_4	F_5	F_6
[15]	LSSS	✓	×	×	×	×	×
[20]	LSSS	✓	×	×	×	×	×
[28]	Tree	×	×	✓	✓	×	×
[25]	LSSS	×	×	✓	✓	×	✓
[27]	LSSS	✓	✓	✓	×	✓	✓
[29]	LSSS	✓	×	✓	✓	×	✓
[14]	Tree	✓	×	✓	✓	✓	×
[30]	LSSS	✓	×	✓	✓	✓	×
[31]	LSSS	✓	✓	✓	✓	✓	×
Ours	LSSS	✓	✓	✓	✓	✓	✓

6 Performance Evaluation

6.1 Functionality

Table 1 demonstrates a functionality comparison between several schemes based on six features. We think that these features are important in each practical scheme. The first feature(F_1) refers to the outsourcing decryption. Except [25, 28], other schemes use outsourced decryption to reduce the computational burden on the user side. An important part of the outsourced decryption is done by a third party which is usually called partial decryption. In some schemes, partial decryption needs a token generation and so, the data users have to do computation and use their communication channels to send this token. On the other hand, in some schemes including the proposed scheme, partial decryption is done without token generation which is a desirable feature. F_2 refers to this feature. The third feature (F_3) is about whether there is a revocation method in the scheme or not. F_4 is about immediate revocation which can be seen in all the schemes except [15, 20, 27]. As we have already mentioned(in Section 1.2) ciphertext size increases as a consequence of the direct revocation method. F_5 is related to this issue. In the proposed scheme, the revocation process is independent of data users. In other words, during the revocation process, users' secret keys will be unchanged. This feature reduces the communication cost on the user side. F_6 refers to this feature. Table 1 shows that the proposed scheme is the only scheme with all of these features.

6.2 Computation Overhead

Before talking about the computational costs, we examine the execution time of different cryptographic operations. We have provided a code for estimating the running time of different cryptographic operations using Charm framework [32] which is a Python-based library. Our code is available on GitHub [33] and uses SS1024 which is a supported supersingular curve in the Charm. We have run our code in Ubuntu 20.04.4 which has been installed on a virtual machine platform VMware@Workstation.Pro 16.2.3 by a 2.4 GHz Intel Core i5 processor with 8GB RAM. The simulation results are shown in Table 2. It is obvious that the first three operations in this table (pairing, scalar multiplication in $\mathbb{G}$, and exponentiation in $\mathbb{G}_T$) are the main costs of computation and the other operations do not have a significant effect on the running time of algorithms. Therefore, for simplicity, we count the number of these operations in the execution of algorithms and ignore the cost of the other operations. Table 3 demonstrates the computational overhead in several schemes based on the number of mentioned operations. Using our simulation results in Table 2, the execution time of each pairing is equal to $T_P = 26.55ms$. Furthermore, the execution time of scalar multiplication and exponentiation are equal to $T_E = 18.75ms, T_{E_T} = 2.22ms$. We apply these times to the relations in Table 3 and plot the computational time of encryption and partial decryption according to the number of attributes. The result are shown in Figure 3 and Figure 4. Figure 3 shows that the proposed scheme and [31] are more efficient than the others in the encryption and Figure 4 shows that with an increase in the number of attributes, there is a significant time difference between the partial decryption in our scheme and the other schemes.

Table 2. Average execution time of cryptographic operations

Operation	Value
Pairing	26.550 ms
Scalar Multiplication in $\mathbb{G}$	18.754 ms
Exponentiation in $\mathbb{G}_T$	2.220 ms
Addition in $\mathbb{G}$	0.017 ms
Multiplication in $\mathbb{G}_T$	0.004 ms
Multiplication in $\mathbb{Z}_p$	0.001 ms
Inverse Operation in $\mathbb{G}$	0.001 ms
Inverse Operation in $\mathbb{G}_T$	0.013 ms

Table 3. Computational overhead comparison. E: Computational overhead of exponentiation in $\mathbb{G}$, E_T: Computational overhead of exponentiation in $\mathbb{G}_T$, P: Computational overhead of pairing, l: Number of attributes used in the encryption, l_d: Number of attributes used in the decryption

Scheme	Encryption (Total Cost)	Partial Decryption (Third-Party Side)	Decryption (User Side)
[15]	$(5l+1)E + E_T$	$(3l_d+2)P + E + (l_d)E_T$	E_T
[20]	$(5l)E + (2l+1)E_T$	$(4l_d)P + (l_d)E_T$	E_T
[27]	$(4l+2)E + E_T$	$(3l_d+2)P + (l_d)E_T$	$(2)P$
[30]	$(3l)E + E_T$	$(2l_d)P + (l_d+1)E + (2l_d)E_T$	E_T
[31]	$(2l+2)E + E_T$	$(l_d+3)P + (l_d)E + (l_d)E_T$	P
Ours	$(2l+2)E + E_T$	$(4)P + (2l_d+2)E$	E_T

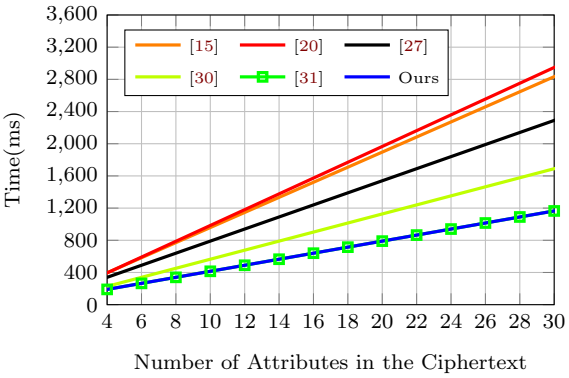


Figure 3. Encryption execution time

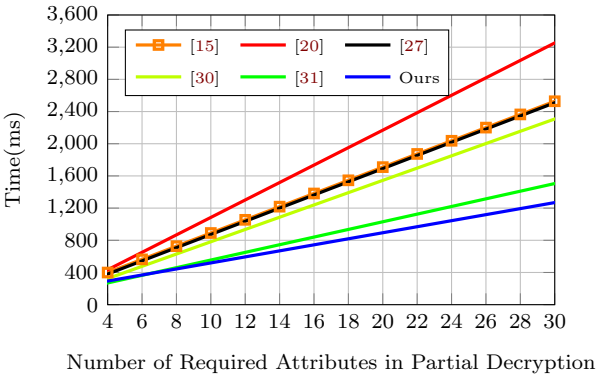


Figure 4. Partial decryption execution time

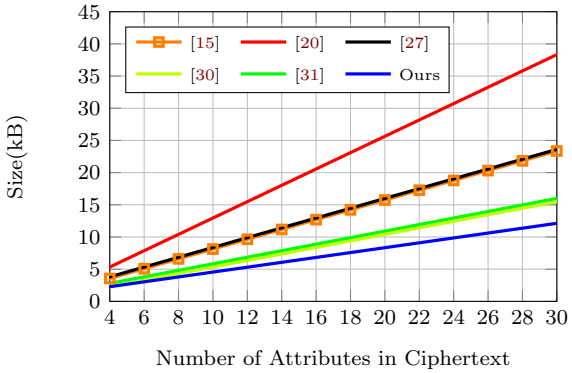


Figure 5. Ciphertext size

6.3 Storage Overhead

Storage overhead can be measured by three factors which are shown in Table 4. The user's secret key, ciphertext, and the public parameter sizes are important factors because these parameters must be stored and also transmit through communication channels. In the elliptic curve SS1024, the size of each field element is equal to $|\mathbb{Z}_p| = 1024 bits$. Furthermore, $|\mathbb{G}| = |\mathbb{G}_T| = 2080 bits$. We apply these sizes to the ciphertext size column in Table 4 and plot the results according to the number of attributes in Figure 5. Based on this figure our scheme has better performance in the ciphertext size.

Table 4 shows that the secret key size in our scheme and [27] are very small and fixed which is an appropriate feature. But in the proposed scheme, the size of public parameters increases with the number of attributes in the system which must be modified in the future works [3].

7 Conclusion

Recently, based on elliptic curve cryptography, several CP-ABE schemes have been proposed. These schemes have much less computational cost than pairing-based schemes but the lack of accurate security proof makes them useless. Our scheme has security proof based on traditional approaches in pairing-based cryptography but uses a construction that is in some ways similar to the elliptic curve cryptography approaches. Similar to elliptic curve-based schemes, in the proposed scheme field $\mathbb{Z}_p$ elements are directly used in the public parameters, ciphertexts, and users' keys. Therefore, some of the decryption computations are done by multiplication in the field $\mathbb{Z}_p$ without the need for pairing. Consequently, our scheme uses only four

Table 4. Storage overhead comparison. $|\mathbb{G}|$: Storage overhead of an element in $\mathbb{G}$, $|\mathbb{G}_T|$: Storage overhead of an element in $\mathbb{G}_T$, $|\mathbb{Z}_p|$: Storage overhead of an element in $\mathbb{Z}_p$, l_k: Number of attributes in the user's key, l: Number of rows in the ciphertext access structure, $|\mathbb{U}|$: Total number of attributes in the system, n_{aa}: Number of attribute authorities

Scheme	Secret Key Size	Ciphertext Size	Public Parameters Size
[15]	$(2l_k + 3)\|\mathbb{G}\|$	$(3l + 1)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$6\|\mathbb{G}\|+2\|\mathbb{G}_T\|$
[20]	$(4l_k + 1)\|\mathbb{G}\|$	$(5l)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(3n_{aa})\|\mathbb{G}\|+(n_{aa})\|\mathbb{G}_T\|$
[27]	$\|\mathbb{G}\|$	$(3l + 2)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(7)\|\mathbb{G}\|+\|\mathbb{G}_T\|$
[30]	$(l_k+2)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(2l)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(3)\|\mathbb{G}\|+(2)\|\mathbb{G}_T\|$
[31]	$(l_k + 3)\|\mathbb{G}\|$	$(2l + 2)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(2)\|\mathbb{G}\|+\|\mathbb{G}_T\|$
Ours	$\|\mathbb{Z}_p\|$	$(l + 2)\|\mathbb{G}\|+\|\mathbb{G}_T\|+(l)\|\mathbb{Z}_p\|$	$(\|\mathbb{U}\|+2)\|\mathbb{G}\|+\|\mathbb{G}_T\|+(\|\mathbb{U}\|)\|\mathbb{Z}_p\|$

[3] Similar to other works, this issue can be corrected using a hash function. This function is modeled as a random oracle in the security proof but for simplicity, we ignore that.

pairings during the decryption process and asymptotically achieves the same computational efficiency as the mentioned elliptic curve-based schemes. We think this approach can be useful in other cryptography primitives. The proposed scheme uses secure outsourced decryption with a user-independent revocation method which makes our scheme suitable for applications such as IoT due to the very low computation and communication overhead on the user side.

References

[1] Diao Zhe, Wang Qinghong, Su Naizheng, and Zhang Yuhan. Study on data security policy based on cloud storage. In *2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids)*, pages 145–149. IEEE, 2017.

[2] Pierangela Samarati and Sabrina Capitani de Vimercati. Access control: Policies, models, and mechanisms. In *International School on Foundations of Security Analysis and Design*, pages 137–196. Springer, 2000.

[3] Amos Beimel et al. Secure schemes for secret sharing and key distribution. 1996.

[4] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE symposium on security and privacy (SP'07)*, pages 321–334. IEEE, 2007.

[5] Yi-Fan Tseng and Jheng-Jia Huang. Cryptanalysis on two pairing-free ciphertext-policy attribute-based encryption schemes. In *2020 International Computer Symposium (ICS)*, pages 403–407. IEEE, 2020.

[6] Yong Wang, Biwen Chen, Lei Li, Qiang Ma, Huicong Li, and Debiao He. Efficient and secure ciphertext-policy attribute-based encryption without pairing for cloud-assisted smart grid. *IEEE Access*, 8:40704–40713, 2020.

[7] Yang Ming, Baokang He, and Chenhao Wang. Efficient revocable multi-authority attribute-based encryption for cloud storage. *IEEE Access*, 9:42593–42603, 2021.

[8] K Sowjanya and Mou Dasgupta. A ciphertext-policy attribute based encryption scheme for wireless body area networks based on ecc. *Journal of Information Security and Applications*, 54:102559, 2020.

[9] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 457–473. Springer, 2005.

[10] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98, 2006.

[11] Ling Cheung and Calvin Newport. Provably secure ciphertext policy ABE. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 456–465, 2007.

[12] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*, pages 53–70. Springer, 2011.

[13] Matthew Green, Susan Hohenberger, Brent Waters, et al. Outsourcing the decryption of abe ciphertexts. In *USENIX security symposium*, volume 2011, 2011.

[14] Mohammad Ali, Javad Mohajeri, Mohammad-Reza Sadeghi, and Ximeng Liu. A fully distributed hierarchical attribute-based encryption scheme. *Theoretical Computer Science*, 815:25–46, 2020.

[15] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Kaitai Liang, Hui Ma, and Lifei Wei. Auditable σ-time outsourced attribute-based encryption for access control in cloud computing. *IEEE Transactions on Information Forensics and Security*, 13(1):94–105, 2017.

[16] Jiguo Li, Yao Wang, Yichen Zhang, and Jinguang Han. Full verifiability for outsourced decryption in attribute based encryption. *IEEE transactions on services computing*, 13(3):478–487, 2017.

[17] Muhammad Asim, Milan Petkovic, and Tanya Ignatenko. Attribute-based encryption with encryption and decryption outsourcing. 2014.

[18] Rui Zhang, Hui Ma, and Yao Lu. Fine-grained access control system based on fully outsourced attribute-based encryption. *Journal of Systems and Software*, 125:344–353, 2017.

[19] Kai Fan, Tingting Liu, Kuan Zhang, Hui Li, and Yintang Yang. A secure and efficient outsourced computation on data sharing scheme for privacy computing. *Journal of Parallel and Distributed Computing*, 135:169–176, 2020.

[20] Kamalakanta Sethi, Ankit Pradhan, and Padmalochan Bera. Practical traceable multi-authority CP-ABE with outsourcing decryption and access policy updation. *Journal of Information Security and Applications*, 51:102435, 2020.

[21] Vanga Odelu, Ashok Kumar Das, Muhammad Khurram Khan, Kim-Kwang Raymond Choo, and Minho Jo. Expressive CP-ABE scheme for mobile devices in IoT satisfying constant-size keys and ciphertexts. *IEEE Access*, 5:3273–3283, 2017.

[22] Xuanxia Yao, Zhi Chen, and Ye Tian. A lightweight attribute-based encryption scheme for the internet of things. *Future Generation Computer Systems*, 49:104–112, 2015.

[23] K Sowjanya, Mou Dasgupta, Sangram Ray, and Mohammad S Obaidat. An efficient elliptic curve cryptography-based without pairing KPABE for internet of things. *IEEE Systems Journal*, 14(2):2154–2163, 2019.

[24] Sheng Ding, Chen Li, and Hui Li. A novel efficient pairing-free CP-ABE based on elliptic curve cryptography for IoT. *IEEE Access*, 6:27336–27345, 2018.

[25] Hao Wang, Zhihua Zheng, Lei Wu, and Ping Li. New directly revocable attribute-based encryption scheme and its application in cloud storage environment. *Cluster Computing*, 20(3):2385–2392, 2017.

[26] Hui Cui, Robert H Deng, Yingjiu Li, and Baodong Qin. Server-aided revocable attribute-based encryption. In *European Symposium on Research in Computer Security*, pages 570–587. Springer, 2016.

[27] Baodong Qin, Qinglan Zhao, Dong Zheng, and Hui Cui. (dual) server-aided revocable attribute-based encryption with decryption key exposure resistance. *Information Sciences*, 490:74–92, 2019.

[28] Jiguo Li, Wei Yao, Jinguang Han, Yichen Zhang, and Jian Shen. User collusion avoidance CP-ABE with efficient attribute revocation for cloud storage. *IEEE Systems Journal*, 12(2):1767–1777, 2017.

[29] Shanshan Tu, Muhammad Waqas, Fengming Huang, Ghulam Abbas, and Ziaul Haq Abbas. A revocable and outsourced multi-authority attribute-based encryption scheme in fog computing. *Computer Networks*, 195:108196, 2021.

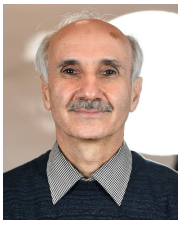
[30] Rui Guo, Geng Yang, Huixian Shi, Yinghui Zhang, and Dong Zheng. O3-R-CP-ABE: An efficient and revocable attribute-based encryption scheme in the cloud-assisted IoMT system. *IEEE Internet of Things Journal*, 8(11):8949–8963, 2021.

[31] Jing Zhao, Peng Zeng, and Kim-Kwang Raymond Choo. An efficient access control scheme with outsourcing and attribute revocation for fog-enabled E-health. *IEEE Access*, 9:13789–13799, 2021.

[32] Joseph A Akinyele, Christina Garman, Ian Miers, Matthew W Pagano, MichaelRushanan, Matthew Green, and Aviel D Rubin. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering*, 3(2):111–128, 2013.

[33] Average execution time of cryptographic operations.https://github.com/Sinaab111/measuring-the-average-execution-time-of-cryptographic-operations.git.

Sina Abdollahi received his B.Sc. and M.Sc. degrees in electrical engineering from Ferdowsi University of Mashhad, Iran, in 2019 and Sharif University of Technology, Iran, in 2022 respectively. His research interests include designing cryptographic primitives and edge computing security.



Javad Mohajeri is an assistant professor with Electronics Research Institute and an adjunct assistant professor with Electrical Engineering Department, Sharif University of Technology, Tehran, Iran. His research interests include the design and cryptanalysis of cryptographic algorithms, protocols, and data security. He is the author/co-author of over 120 research articles in refereed journals/conferences. He has authored 3 books and is one of the founding members of the Iranian Society of Cryptology.

Mahmoud Salmasizadeh received the B.Sc. and M.Sc. degrees in electrical engineering from Sharif University of Technology, Tehran, Iran, in 1972 and 1989, respectively. He also received a Ph.D. degree in information technology from Queensland University of Technology, Australia, in 1997. Currently, he is an associate professor in the Electronics Research Institute and an adjunct associate professor in the Electrical Engineering Department, Sharif University of Technology. His research interests include the design and cryptanalysis of cryptographic algorithms and protocols, e-commerce security, and information-theoretic secrecy. He is a founding member of the Iranian Society of Cryptology.