International Journal of Mathematics Trends and Technology
ISSN: 2231-5373/ https://doi.org/10.14445/22315373/IJMTT-V72I3P109
Volume 72 Issue 3, 71-89, March 2026

Original Article

An Efficient Revocable Attribute-Based Encryption Scheme with Version-Based Revocation and Outsourced Decryption

Bin Ge[1], Yi Chen[1], Chanyi Gong[1], Gang Qiang Duan[2], Chungen Xu[1*], Zhaojie Bu[1], Jinyan Cui[1]

[1] *School of Mathematics and Statistics, Nanjing University of Science and Technology.*
[2]*School of Cyber Science and Engineering, Nanjing University of Science and Technology.*

**Corresponding Author : xuchung@njust.edu.cn*

Received: 23 January 2026 Revised: 27 February 2026 Accepted: 18 March 2026 Published: 29 March 2026

Abstract *- With the rapid advancement of cloud computing and the Internet of Things (IoT), massive data outsourcing has become an inevitable trend. However, this paradigm brings significant challenges to data security and fine-grained access control. Traditional single-authority Attribute-Based Encryption (ABE) schemes suffer from inherent drawbacks such as key escrow, heavy computational overhead, and inefficient user revocation. To overcome these limitations, this paper presents an efficient Multi-Authority Revocable Attribute-Based Encryption (MA-RABE) scheme. The proposed scheme eliminates the need for a centralized authority by employing a distributed key generation protocol, ensuring secure and decentralized master key distribution while mitigating single points of failure. Furthermore, a lightweight attribute revocation mechanism is designed, which only requires updating the user's attribute key and one ciphertext component, significantly reducing computation and communication costs. To guarantee forward security, a version-based revocation strategy is introduced to prevent revoked users from decrypting previously accessible ciphertexts. In addition, the scheme supports outsourced decryption, allowing computationally intensive operations to be executed by a semi-trusted server, thereby minimizing local computational load on end devices. Security analysis shows that the proposed scheme achieves IND-CPA security and resists collusion attacks, while performance evaluation demonstrates that it achieves lower computational overhead in key generation, encryption, and decryption compared with existing approaches. Hence, the scheme is particularly suitable for resource-constrained mobile and IoT environments.*

Keywords *- Multi-Authority, Revocation, Outsourced Decryption, Forward Security, Cloud Security.*

1. Introduction

With the rapid growth and widespread adoption of cloud computing, big data, and the Internet of Things (IoT), more and more enterprises and individuals are choosing to outsource data storage and computation [1][2]. This approach takes advantage of the elastic resources offered by cloud services, greatly improving data processing efficiency while lowering costs. However, it also creates a separation between data ownership and management, which brings serious security concerns. When sensitive data is hosted in external environments, it becomes more vulnerable—whether to internal threats, external attacks, or untrusted service providers—posing significant risks to its confidentiality, integrity, and controllability [3].

To enable secure sharing of outsourced data and achieve fine-grained access control, considerable research efforts have been undertaken in academia and industry. Among various cryptographic primitives, Attribute-Based Encryption (ABE) has emerged as a highly promising solution [4]. The core idea of ABE is to treat user identity as a set of attributes and bind decryption capabilities to access policies. Data owners can define flexible access policies embedded either within ciphertexts (Ciphertext-Policy ABE, CP-ABE) or user keys (Key-Policy ABE, KP-ABE), enabling one-to-many encryption and fine-grained access control without continuously relying on a trusted central party [6][7]. This makes ABE an ideal candidate for building secure data sharing systems. But traditional ABE schemes—especially CP-ABE—still face a problem when deployed in real-world cloud and IoT environments: most schemes rely on a single Central Authority (CA) to handle all key generation and management. This does not just create a trust bottleneck and a single point of failure; it also puts increasing pressure on performance as the system scales up [10].

Sahai and Waters [1] introduced Fuzzy Identity-Based Encryption (Fuzzy IBE), which supports decryption under a certain error tolerance using "fuzzy identities" and laid the groundwork for Attribute-Based Encryption (ABE). In 2006, Goyal et al. [4]further proposed an ABE scheme supporting access policies and categorized ABE into Ciphertext-Policy ABE (CP-ABE) and Key-Policy ABE (KP-ABE) according to where the access structure is embedded. Chase et al.[5] were the first to propose multi-authority attribute-based encryption. This distributed the management of the attribute space among multiple independent authorities, enhancing system robustness and scalability [11]. Chang et al. [8] proposed a T-ABE scheme that migrates the key processing procedure of the ABE scheme to a Trusted Execution Environment (TEE), ensuring the security of this procedure when deployed on a fully untrusted cloud platform. Gudipati et al. [9] put forward a quantum-resistant, traceable, revocable, and non-key-escrow CP-ABE scheme for cloud storage scenarios. However, in dynamic systems, user attributes tend to change frequently, so having an efficient revocation mechanism becomes critical. Existing approaches often require either the data owner or the cloud server to fully re-encrypt the ciphertexts, or they depend on complex key update protocols—both of which bring heavy computational and communication overhead [11].

In current research, within the context of multi-authority ABE, relatively more secure and fully decentralized encryption schemes are scarce. Building upon scheme[8], this paper introduces algorithmic improvements to propose an efficient, decentralized, and revocable attribute-based encryption scheme supporting multiple authorities. By introducing Distributed Key Generation (DKG) in a multi-authority setting, this scheme achieves more secure and fully decentralized encryption. To address the above challenges, the contributions of this paper are as follows:

1. This design adopts a fully decentralized architecture, doing away with the need for any centralized coordination authority. Each attribute authority operates independently, managing its own dedicated domain and handling key generation and permission verification through distributed collaboration. As a result, single-point dependencies are completely eliminated, and the system becomes more robust and scalable.
2. A lightweight attribute revocation mechanism is proposed. The revocation process requires updating the user's attribute key and one ciphertext component, thus avoiding complete data re-encryption. Meanwhile, forward security is ensured to prevent revoked users from accessing data using old keys.
3. The scheme adopts an online/offline approach, offloading computationally intensive operations in the decryption process—such as bilinear pairing and matrix operations—to an offline outsourced server, while the user terminal only needs to perform simple exponential operations. This effectively reduces resource consumption on end devices, making the scheme well-suited for lightweight devices like mobile terminals.

The rest of this paper is organized as follows: Section 2 reviews related work. Section 3 introduces cryptographic preliminaries. Section 4 presents the system model and scheme overview. Section 5 details the construction of our scheme. Section 6 provides a security analysis. Section 7 evaluates performance, and Section 8 concludes the paper.

2. Related Works

This section is dedicated to related work on Multi-Authority Attribute-Based Encryption (MA-ABE), revocation mechanisms, and outsourced cryptographic operations.

2.1. Multi-Authority Attribute-Based Encryption

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) has been widely adopted in various cloud scenarios due to its fine-grained access control capabilities, which allow data owners to define access policies flexibly. However, single-authority ABE schemes suffer from issues such as excessive computational burden, high risk of single-point failure, and over-reliance on a central authority. To mitigate these limitations, Chase [5] introduced a Multi-Authority ABE (MA-ABE) scheme. Subsequently, Chase et al.[13] further enhanced user privacy by eliminating the trusted central authority, thereby preventing excessive concentration of user information. Lin et al. [14] proposed a threshold-based Multi-Authority Fuzzy Identity-Based Encryption (MA-FIBE) scheme, which can be extended into an MA-ABE framework. Liu et al. [15] presented an MA-ABE scheme with constant-size ciphertexts and support for partially hidden policies. Wu et al. [27] proposed a multi-authority attribute-based security scheme featuring an updatable policy. By introducing a policy update key, the scheme enables efficient access to policy updates with low overhead.

While Multi-Authority ABE (MA-ABE) effectively disperses single-point bottlenecks and enhances privacy, it still faces three key challenges: the communication overhead of inter-authority coordination, a lack of safeguards against semi-trusted authorities, and scalability limitations with a growing number of entities.

2.2. Revocable Attribute-Based Encryption

Hur et al. [3] integrated ABE with group management techniques to achieve revocation. However, a revoked user may still access outsourced data using other valid attributes they retain. Hoang et al. [6] proposed a CP-ABE scheme that supports both

attribute and user revocation. By incorporating re-encryption techniques, their approach ensures the confidentiality of historical communications and employs a binary tree structure for efficient user information storage. Xiang et al. [16] designed a real-time, revocable, fine-grained attribute-based CP-ABE scheme that leverages version control to enable instant attribute revocation. Guo et al. [11] proposed an efficient traceable and revocable ABE scheme for cloud storage environments. Ge et al. [28] proposed an attribute-based proxy re-encryption with direct revocation (ABPRE-DR) for encrypted data sharing, which enables cloud servers to directly revoke the original shared set of users associated with the re-encryption key. Yin et al. [7] introduced a traceable CP-ABE scheme with direct revocation tailored for medical data sharing. Zhou et al. [17] utilized broadcast encryption to revoke user keys; however, their scheme only supports a single authority and does not address the key escrow problem.

Revocation is crucial for the practical deployment of ABE. While direct and indirect revocation suffer from limitations in thoroughness and real-time performance, respectively, existing schemes generally face an efficiency-security trade-off and are difficult to integrate into multi-authority architectures. Consequently, achieving secure and efficient real-time revocation in a decentralized setting remains an open challenge.

2.3. Outsourced Encryption and Decryption

To address the issue of high computational overhead in traditional ABE for resource-constrained clients, Khan et al. [18] introduced an online/offline-assisted attribute-based multi-keyword search scheme, which shifts the most computationally intensive tasks to the offline phase, effectively reducing online computation overhead. Li et al.[19] designed an online/offline MA-CP-ABE scheme based on cryptographic reverse firewalls for IoT scenarios. For applications requiring high security and real-time performance, Luo et al.[20] proposed an efficient CP-ABE scheme in industrial settings that offloads most computations to servers while supporting policy hiding. Li et al. [29] proposed a flexible policy-hiding multi-group attribute-based encryption (PH-MG-ABE) scheme. To alleviate the local decryption burden on users, the heavy decryption tasks are outsourced to a cloud server, and the correctness of the outsourced decryption is verifiable.

Outsourcing strategies enable ABE on resource-limited devices by offloading intensive computations, with online/offline cryptography further boosting efficiency. However, these approaches inherently depend on the cloud server's honesty, requiring verification mechanisms and introducing privacy risks from delegated keys/tasks. Moreover, research lags in securely outsourcing the encryption phase, especially within the challenging multi-authority setting.

Research has made significant yet siloed advancements in MA-ABE, revocation, and computation outsourcing. A holistic solution that integrates a multi-authority architecture, efficient revocation, and verifiable computation outsourcing remains elusive. Thus, an innovative scheme is urgently needed to build a practical and lightweight ABE system for resource-constrained users.

3. Preliminaries

3.1. Bilinear Pairing

Consider three multiplicative cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ of prime order r. Let $\mathbb{G}_1 = \langle g_1 \rangle$ and $\mathbb{G}_2 = \langle g_2 \rangle$, where g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. A bilinear pairing is a map $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties [21]:

1. *Bilinearity:* for all , $d \in \mathbb{Z}_r$, $e(g_1^c, g_2^d) = e(g_1, g_2)^{cd}$.
2. *Non-degeneracy:* The map e does not send the generators g_1 and g_2 to the identity element of $\mathbb{G}_T$, i.e.,$e(g_1, g_2) \neq 1_{\mathbb{G}_T}$.This implies that for any generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, the value $e(g_1, g_2)$ is also a generator of $\mathbb{G}_T$.
3. *Efficiently computable:* for any $p \in \mathbb{G}_1$, and $q \in \mathbb{G}_2$, there exists an efficient algorithm to calculate $e(p, q) \in \mathbb{G}_T$.

3.2. Linear Secret Sharing Scheme (LSSS)

Consider a set of parties P and a secret sharing scheme Π over P. Π is called a linear secret sharing scheme over $\mathbb{Z}_r$ if Π fulfills the following requirements [22]:

1. Every party $P \in P$ holds a piece of information called a share, and the collection of all shares forms a vector over $\mathbb{Z}_r$.

2. There exists a share-generating matrix A for Π of size $(l \times n)$, along with a labeling function $\rho(i): \{1, \dots, l\} \to$ P that assigns each row i of A to a specific party. Let $s \in \mathbb{Z}_r$ be a secret that will be shared using Π , and $v = (s, s_2, \dots, s_n)$ be a column vector, where $s_2, \dots, s_n \in \mathbb{Z}_r$ are chosen uniformly at random. In addition, for all $i \in \{1, \dots, l\}, \lambda_i = A_i v$ is the vector of l shares that distribute a secret, denoted s , according to the scheme Π . The share λ_i belongs to the party indicated by $\rho(i)$. In the context, attributes serve the role of the parties described above.

3.3. Security Assumption

Definition 1 (Decisional Bilinear Diffie-Hellman (DBDH) in Type-3 Pairings Assumption). Given a valid Type-3 pairing defined on the set of parameters $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, r\}$, the DBDH assumption holds if no probabilistic polynomial-time PPT algorithm can distinguish a DBDH tuple $\langle g_1^a, g_1^b, g_1^c, g_2^a, g_2^b, g_2^c, e(g_1, g_2)^{abc} \rangle$ from a random tuple $\langle g_1^a, g_1^b, g_1^c, g_2^a, g_2^b, g_2^c, e(g_1, g_2)^z \rangle$ with nonnegligible advantage, where $a, b, c, z \in \mathbb{Z}_r, g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ are selected uniformly at random. The problem of distinguishing a DBDH tuple from a random tuple is called the DBDH problem. [10]

4. System Model and Scheme Overview

4.1. System Model

Instead of relying on a Central Trusted Authority (CTA), the proposed scheme distributes trust and functional responsibilities across five core entities (Fig), namely the Data Owner (DO) and the Attribute Authorities (AAs). As shown in Fig, the interactions among these Attribute Authorities, the Outsource Service Provider (OSP), the Cloud Service Provider (CSP), the Data Owner, and the users together form the overall system architecture.

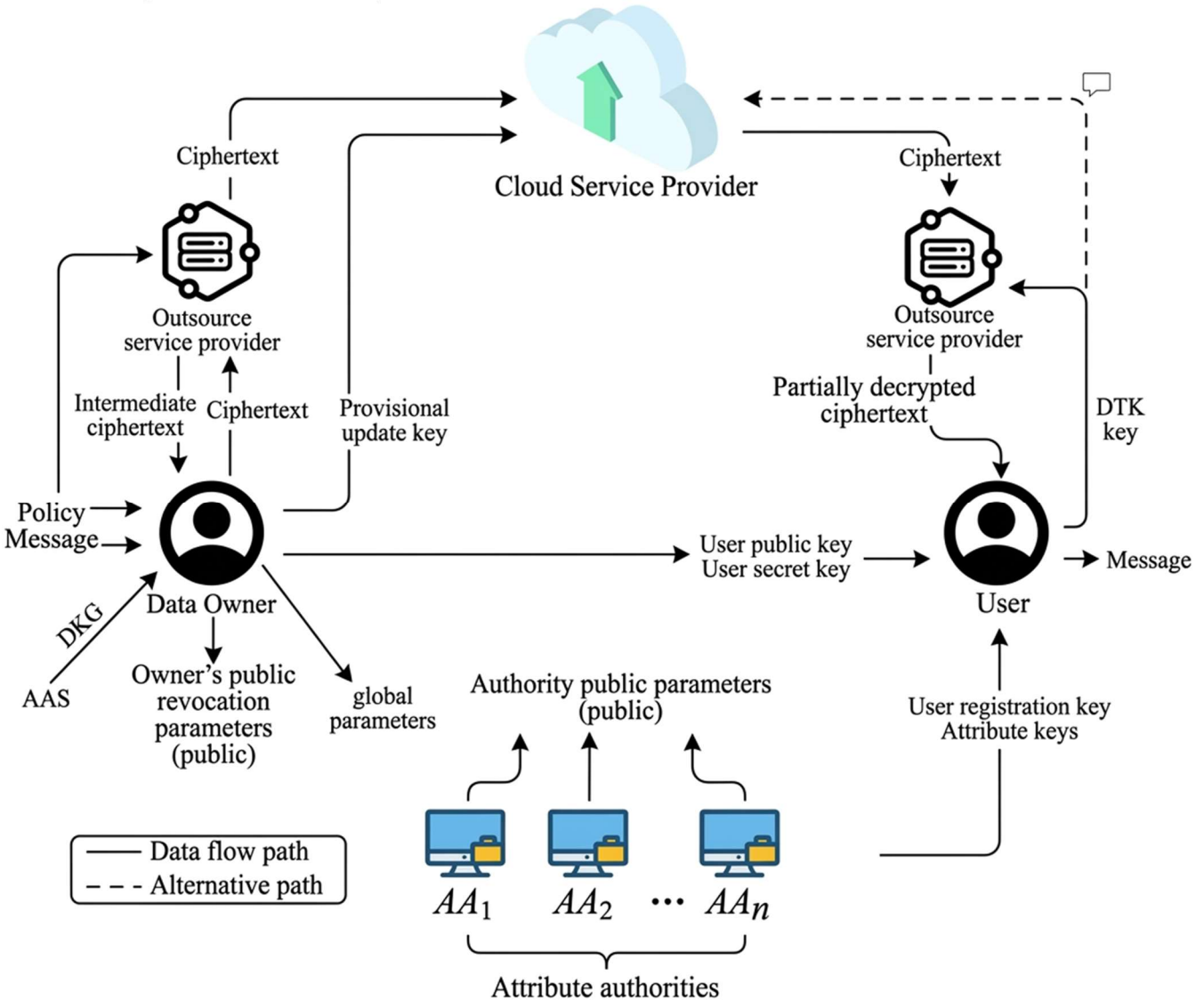

Fig. 1 System model

1. ***Attribute Authorities (AAs):*** These entities are responsible for managing disjoint sets of attributes (e.g., AA1 manages attributes like "Engineer," while AA2 manages attributes like "Location=Factory1"). They collaboratively generate the global system parameters through a Distributed Key Generation (DKG) protocol, handle user registration, and issue attribute-specific secret keys. Critically, no single AA possesses full control over the system.

2. ***Outsource Service Provider (OSP):*** Acting as a semi-trusted entity deployed between IoT devices and the cloud, the OSP performs computationally intensive encryption and decryption operations on behalf of resource-constrained devices. It also verifies the validity of user attribute keys during the decryption process.
3. ***Cloud Service Provider (CSP):*** The CSP is responsible for storing ciphertexts, public parameters, and keys associated with the DO. When a revocation event occurs, the CSP updates the affected ciphertexts using update keys provided by the DO.
4. ***Data Owners (DOs):*** The Data Owner (DO) encrypts the data with a selected version number, enforces access control by encrypting the key under a specified policy, attaches a revocation component, and finally uploads the resulting ciphertext to the Cloud Storage (CSP).
5. ***Users (Us):*** These are resource-constrained IoT devices (e.g., sensors, actuators) that request and consume encrypted data. They generate transformation keys for the OSP to enable outsourced decryption and subsequently use their local recovery keys to obtain the final plaintext.

4.2. Algorithm Definition

Table 1 shows the relevant symbols used in this section.

Table 1. Description of symbols used in the document

Symbol	Description	Symbol	Description
λ	Security level	$D_{uid,x}$	Attribute x key of uid
n	Number of users (maximum)	DTK_{uid}	Decryption Transformation Key of uid
GP	Global parameters	z_{uid}	Recovery key of uid
AID	Set of AAs identifiers	u_{oid}	Owner secret revocation key of oid
Aid，uid，oid	Authority, user, and owner identifiers, respectively.	SR	The secret revocation component of oid
S_{aid}	Set of attributes of AA aid	PR_{oid}	Public revocation parameters of oid
PP_{aid}	Public parameters of AA aid	PUK_{oid}	Provisional update key of oid
MSK_{aid}	Master key of AA aid	A	Access policy
PPR	Public parameters of revocation	IC	Intermediate ciphertext
SKR	Secret key for revocation	m	Message
CR	Revocation component	CT	Ciphertext
UPK_{uid}	User public key of uid	$\overline{CT}$	Partially decrypted ciphertext
USK_{uid}	User secret key of uid	URK_{oid}	Update the revocation key of oid
$K_{uid.aid}$	User registration key of uid with aid	CT′	Updated ciphertext (after revocation)

The scheme consists of 15 algorithms grouped into five phases, with CTA functionalities replaced by AA-DO collaboration. As depicted in Fig, these phases are the setup phase, the key generation phase, the encryption phase, the decryption phase, and the revocation phase.

During the setup phase, both the Attribute Authority (AA) and the Data Owner (DO) collaboratively generate the global system parameters and master keys. In the key generation phase, the AA distributes attribute keys to users. In the encryption phase, the data owner encrypts the data according to the predefined access policy and outsources the ciphertext to the cloud.

During the decryption phase, the Outsourced Service Provider (OSP) assists users with partial decryption to reduce their computational overhead. In the revocation phase, the system efficiently enforces access control for revoked users by updating the revocation components and ciphertexts. The algorithms included in the proposed scheme are outlined below.

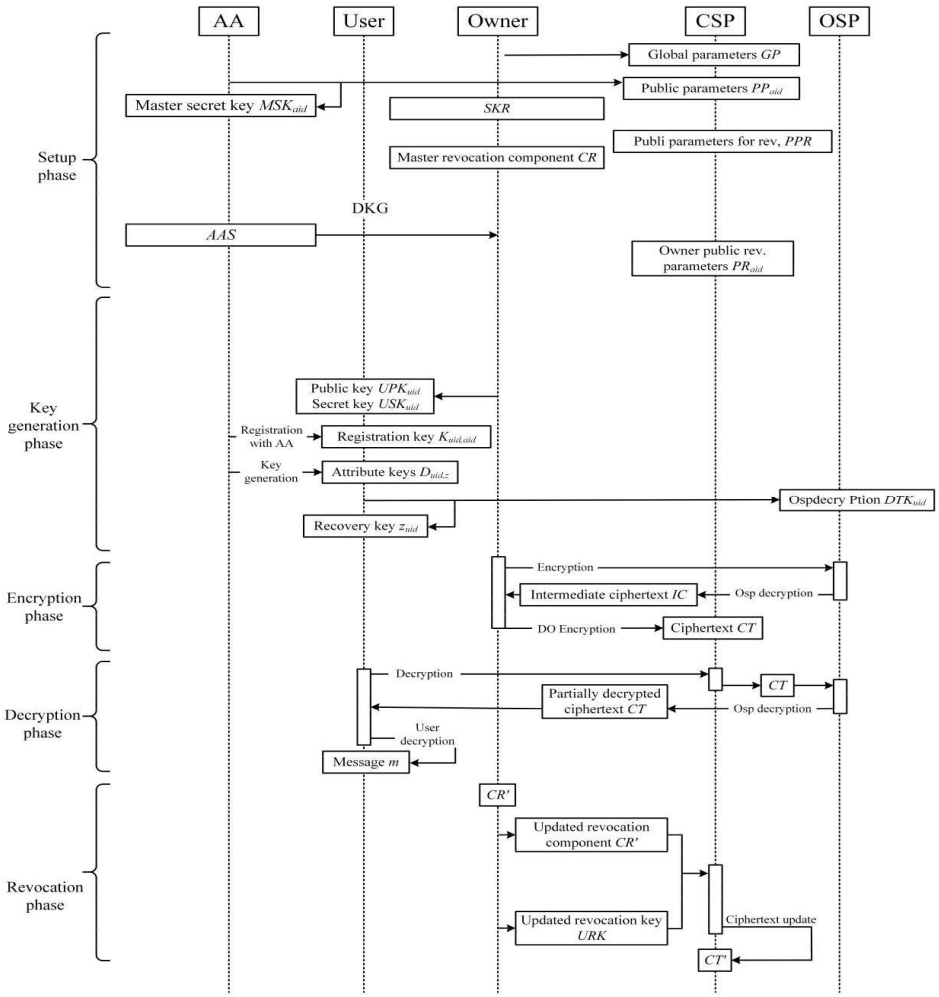

Fig. 2 Workflow of the proposed scheme

(1) Setup Phase

DistributedSetup $(\lambda, N_{aa}, t) \rightarrow (GP)$: Based on the security parameters λ and the parameters obtained from running the Pedersen DKG protocol, this algorithm is executed by the DO to generate the global parameters GP . The GP will include the set of N_{aa} 's identifiers AID and the hash function H .

- ***AASetup***$(GP, aid, S_{aid}, t_{aid}) \rightarrow (PP_{aid}, MSK_{aid})$ *:* This algorithm is executed by the AA. It takes as inputs the global parameters P , the AA identifier aid, and the attribute set S_{aid} , and the local secret t_{aid} obtained from the DKG key distribution. It outputs the AA's public parameters PP_{aid} and the AA's master secret key MSK_{aid} .

- ***RevSetup*** $(GP, n, \alpha, u) \rightarrow (PPR, SKR)$: This algorithm is executed by the DO. It takes as inputs the GP, the maximum number of system users n , the master secret key α obtained from DKG key distribution, and the initial version number u . It outputs the public parameters PPR and the secret key for revocation SKR held by the DO.

(2) Key Generation Phase

- ***UserRegAADistributed*** $(GP, PPR, SKR) \rightarrow (UPK_{uid}, PK_{uid})$: In this algorithm, the DO handles user registration by assigning each user a unique identity identifier and generating the corresponding keys. These keys play a critical role—they are used for both data decryption and attribute revocation. Specifically, the algorithm outputs a user identifier as uid, along with the user's public key as UPK_{uid} and private key PK_{uid}.

- ***UserRegAA*** $(GP, MSK_{aid}, UPK_{uid}) \rightarrow (K_{uid,\,aid})$:This algorithm is executed by the AAs to register each user. It takes the GP, MSK_{aid} and UPK_{uid} as inputs and generates a user registration key. This key serves as proof that the user has completed registration with the specific Attribute Authority.

- $\boldsymbol{KeyGen}(GP, S_{uid,aid}, MSK_{aid}, UPK_{uid}, u) \rightarrow \left(\{D_{uid,x}\}_{x \in S_{uid,aid}}\right)$: This algorithm is executed by each AA to generate version-bound attribute keys for users who have previously registered with the AA. It takes as inputs the GP, MSK_{aid}, UPK_{uid} , and the set of attributes of uid managed by aid, denoted $S_{\text{uid , aid}}$, and the version number u obtained from the DO. It outputs a set of attribute keys $\{D_{\text{uid},x}\}_{x \in S_{\text{uid , aid}}}$. These keys are private and will be used to generate OSP keys.

- $\boldsymbol{UserKeyGen_{out}}(USK_{uid}, K_{uid}, D_{uid}) \rightarrow (DTK_{uid}, Z_{uid})$ *:* Each user executes this algorithm to generate OSP keys required for outsourced decryption. The algorithm takes as inputs USK_{uid} , K_{uid} and the user's version-bound attribute keys $D_{uid} = \{D_{uid,x}\}_{x \in S_{uid,aid}}$, and outputs the required OSP keys and user recovery keys Z_{uid} . These keys must be kept private by the user.

(3) Encryption Phase

- ***OSPEncrypt*** $(GP, PP_{AID}, \mathbb{A}) \rightarrow$ *(IC)*: An OSP (Outsource Service Provider) runs the OSP encryption algorithm, initiated by a DO. It receives GP, the public parameters of the AAs PP_{AID} , and an access policy $\mathbb{A}$ as input. This process is executed before the DO generates the data to be encrypted. The algorithm outputs the intermediate ciphertext IC that is returned to the DO.

- ***DOEncrypt*** $(GP, IC, \mathbb{A}, m, SKR, PP_{AID}) \rightarrow (CT)$: The encryption algorithm is run by the DO. It takes GP, IC, $\mathbb{A}$, a message m , SKR, and PP_{AID} as input, the algorithm generates the final ciphertext CT , which is associated with the current version number. The CT is directly uploaded to the CSP.

(4) Decryption Phase

- $\boldsymbol{OSPDecrypt}(GP, CT, DTK_{uid}, u, RL) \rightarrow (\overline{CT})$: The OSP decryption algorithm is executed by an OSP to perform an outsourced partial decryption operation. It intakes GP, CT, FK_{uid} , and the current version number u . First, it verifies whether the version number of the user's key matches that of the ciphertext. If they match, it then checks whether the user's attribute set S_{uid}/RL satisfies the access policy in CT. If both conditions are met, the algorithm outputs the partially decrypted ciphertext. $\overline{CT}$. If the version numbers do not match or the attribute set does not satisfy the policy, the algorithm terminates. The $\overline{CT}$ is delivered to the corresponding user who requested the ciphertext.

- ***UserDecrypt*** $\left(GP, PPR, \overline{CT}, USK_{uid}, z_{uid}\right) \rightarrow (M)$:This algorithm is executed by the user. It takes GP, PPR, $\overline{CT}, USK_{uid}$, and z_{uid} as input. In this algorithm, the user decrypts using their identity secret key and the revocation component. Then, the algorithm returns the decrypted message M .

(5) Revocation Phase

- ***DORevoke*** *(GP, SKR, PPR,* $u_0, RL) \rightarrow$ *(SKR', PPR')*: The DO executes the user's attribute revocation algorithm. It takes GP, SKR, PPR, and the new version number u_0 , and the set RL of users or user attributes to be revoked as input. The algorithm outputs the updated Public Revocation Parameters PPR' and the updated Secret Revocation Key SKR'.

- ***AARevoke*** $(D_{uid}, RL, u_0, GP) \rightarrow (D'_{uid})$:Each AA executes this algorithm. It takes the updated version number u_0 obtained from the DO and GP as input. For users not in RL, it updates their attribute keys associated with the version to D'_{uid} .

- ***CTUpdate*** $(GP, CT, CR') \rightarrow (CT')$:The CSP executes this algorithm. It takes GP, the CT, and the updated revocation component CR' received from the DO as input. Using these, it generates a CT' to replace the CT stored in the CSP.

4.3. Security Definition

IND-CPA: In this setup, we consider a Probabilistic Polynomial-Time (PPT) algorithm $\mathcal{A}$ as the adversary, and let $\mathcal{B}$ denote the challenger. The security of the proposed attribute-based encryption scheme against Chosen-Plaintext Attacks (CPA) can then be captured through the following game played between the two.

Initialization: The adversary $\mathcal{A}$ submits the access policy A to be attacked to the challenger $\mathcal{B}$.
Setup: The challenger $\mathcal{B}$ takes the security parameter λ as input, executes the DistributedSetup algorithm to generate the Global Parameters (GP), and sends the global parameters GP to the adversary .

***Phase 1**:* The adversary $\mathcal{A}$ queries the challenger $\mathcal{B}$ for the attribute key corresponding to any attribute set S, where S does not satisfy the access policy A; the challenger $\mathcal{B}$ executes the KeyGen algorithm and sends the attribute key corresponding to the attributes in the attribute set S to the adversary $\mathcal{A}$.

Challenge: The adversary $\mathcal{A}$ selects two messages of the same length $(\mathrm{m}_0, \mathrm{m}_1)$ and sends them to the challenger $\mathcal{B}$; then the challenger randomly selects a bit $b \in \{0\ ,\ 1\}$, runs the DoEncrypt algorithm with the input message m_b and A , and obtains the ciphertext CT_b . Next, CT_b is sent to the adversary.

Phase 2: For different attribute sets $\mathbf{S}'$, the adversary $\mathcal{A}$ can request more keys, with the only restriction that S′ does not satisfy the policy A. The challenger $\mathcal{B}$ delivers the requested attribute keys to the adversary $\mathcal{A}$.

Guess: Finally, the adversary $\mathcal{A}$ guesses $b' \in \{0,1\}$. If $b' = b$, the adversary wins the above game. The advantage of the adversary $\mathcal{A}$ in winning the above game is $Adv(\mathcal{A}) = |Pr(b' = b) - 1/2|$.

If the advantage $Adv(\mathcal{A})$ on winning the described security game is negligible for any PPT adversary $\mathcal{A}$, it is said that an ABE scheme is secure against chosen-plaintext attacks or CPA-secure.

5. Concrete Construction

This section presents the detailed definition of the proposed Efficient Revocable Attribute-Based Encryption Scheme with Version-Based Revocation and Outsourced Decryption.

5.1. Detailed Algorithms

5.1.1. Setup Phase

1. DistributedSetup $(\lambda, N_{aa}, t) \rightarrow (GP)$ *:*

- According to the inputted security parameter λ , define the pairing $e : G_1 \times G_2 \rightarrow G_T$, where $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$, and G_T have prime order r.
- K AAs set the threshold t , run Pedersen DKG, and obtain the local secret t_{aid} . (aid, $\alpha_{i,o}$) is sent to DO through a private channel, where aid is the identifier defined by AA, and $\alpha_{\mathrm{i,o}}$ is the constant coefficient obtained in the DKG process.
- DO outputs the set of global parameters as:

$$GP = \left(G_1, G_2, G_T, g_1, g_2, e, H, g_2^{\alpha} = g_2^{\sum_k \alpha_{i,o}}, AID\right)$$

The DO publishes GP, which will be available for all the system entities.

2. $\boldsymbol{AASetup}(GP, aid, S_{aid}, t_{aid}) \rightarrow (PP_{aid}, MSK_{aid})$: $\mathrm{S_{aid}}$ represents the attributes controlled by the authority. If $aid_1 \neq aid_2$, then $S_{aid_1} \cap S_{aid_2} = \varnothing$.

- Randomly select exponents $\alpha_{aid}, \beta_{aid} \in Z_r$, and compute $g_1^{\alpha_{aid}}$, $e(g_1, g_2)^{\alpha_{aid}}$, $g_1^{\beta_{aid}}$
- For each attribute $x \in S_{aid}$, select a random exponent t_x , and compute the public key for attribute x as $T_x = g_1^{t_x \cdot t_{aid}}$.
- Output the public parameters and the master secret key for aid:

$$PP_{aid} = \left(e(g_1, g_2)^{\alpha_{aid}}, g_1^{\beta_{aid}}, \{T_x\}_{\forall x \in S_{aid}}\right)$$

$$MSK_{aid} = \left(g_1^{\alpha_{aid}}, \beta_{aid}, t_{aid}, \{t_x\}_{\forall x \in S_{aid}}\right)$$

AA_{aid} publishes its public parameters PP_{aid} while keeping MSK_{aid} confidential.

3. RevSetup $(GP, n, \alpha, u) \rightarrow (PPR, SKR)$: The DO runs this algorithm to initialize the revocation mechanism.

- *For* $i \in \{1,2,\cdots,n,n+2,\cdots,2n\}$, *compute* $h_i = g_1^{\alpha^i}, KR = g_1^{\alpha^{n+1}}$.
- *For* $i \in \{1,2,\cdots,n\}$, *compute* $h_i' = g_2^{\alpha^i}$.
- *Select* $\eta \in Z_r$, *and compute* $V_R = g_1^{\eta}$.
- *Select the initial version number* $u \in Z_r$, *and set* $CR = e(KR, g_2)^u = e(g_1, g_2)^{u\alpha^{n+1}}$.

- *Compute* $OCR_0 = g_2^u, OCR_1 = (V_R \cdot \prod_{i \in S} h_{n+1-i})^u$.
- *Define the public parameters PPR and private key SKR prepared for the revocation process:*

$$PPR = \left(\{h_i\}_{i \in \{1,2,\cdots,n,n+2,\cdots,2n\}}, \{h_i'\}_{i \in \{1,2,\cdots,n\}}, OCR_0, OCR_1\right)$$

$$SKR = (\eta, u, V_R, CR, KR)$$

- *Output PPR and SKR. Then, the DO publishes PPR while keeping SKR confidential.*

5.1.2. Key Generation Phase

1. UserRegAADistributed $(GP, PPR, SKR) \rightarrow (UPK_{uid}, PK_{uid})$ *:* When a User applies to join the system, the DO performs the following steps:

- Define uid as the next available index $i \in \{1,2,\cdots,n\}$.
- Randomly select δ_{uid} , then compute $PK_{uid} = g_1^{\delta_{uid}}$ *and* $P_{uid} = g_2^{\delta_{uid}}$.
- Define $R_{uid} = h_i^{\eta} = g_1^{\alpha^i \cdot \eta}$ *for* $uid = i$.
- Output the user's public key and user private key:

$UPK_{uid} = (uid, PK_{uid}), USK_{uid} = (P_{uid}, P_{uid})$ Finally, UPK_{uid} is made public, and the USK_{uid} is sent to Us_{uid} through a secure channel.

2. $\boldsymbol{UserRegAA}(GP, MSK_{aid}, UPK_{uid}) \rightarrow \left(K_{uid,\,aid}\right)$ *: Compute*

$$K_{uid,aid} = g_1^{\alpha_{aid}} \cdot PK_{uid}^{\beta_{aid}} = g_1^{\alpha_{aid}} \cdot g_1^{\delta_{uid} \cdot \beta_{uid}},$$

Then, send $\text{K}_{\text{uid,aid}}$ to Us_{uid} through a secure channel. After the Us_{uid} completes registration with all AAs, set $\text{K}_{\text{uid}} = \{\text{K}_{\text{uid , aid}}\}_{\text{aid} \in \text{AID}}$.

3. $\boldsymbol{KeyGen}\left(GP, S_{uid,\,aid}, MSK_{aid}, UPK_{uid}, u\right) \rightarrow \left(\{D_{uid,x}\}_{x \in S_{uid,\,aid}}\right)$ *:* Set the user's attribute set S_{aid} , and define $S_{uid,aid} = S_{uid} \cap S_{aid}$, which are the user's attributes managed by the authority aid.

- For each attribute $x \in S_{uid\,,\,aid}$, compute the attribute key associated with version number u: $D_{uid,x} = PK_{uid}^{t_{aid} \cdot t_x \cdot H(x)H(u)} = g_1^{\delta_{uid} \cdot t_{aid} \cdot t_x \cdot H(x)H(u)}$.
- Output the set of attribute keys $\{D_{uid,x}\}_{x \in S_{uid,uid}}$. Then, the attribute keys are sent to Us_{uid} through a secure channel. After Us_{uid} receives all attribute keys corresponding to the attributes in S_{uid} , set $D_{uid} = \{D_{uid\,,x}\}_{x \in S_{uid}}$.

4. $\boldsymbol{UserKeyGen_{out}}(USK_{uid}, K_{uid}, D_{uid}) \rightarrow (DTK_{uid}, Z_{uid})$ *:*

- D_{uid} randomly selects $Z_{uid} \in Z_r$. For each $\text{x} \in \text{S}_{\text{uid}}$, compute: $\overline{D}_{uid,x} = D_{uid,x}^{1/Z_{uid}} = g_1^{\delta_{uid} \cdot t_{aid} \cdot t_x \cdot H(x)H(u)/Z_{uid}}$.
- For each $\text{aid} \in \text{AID}$, compute: $\overline{K}_{uid,\,aid} = K_{uid,\,aid}^{1/Z_{uid}} = \left(g_1^{\alpha_{aid}} \cdot g_1^{\delta_{uid} \cdot \beta_{aid}}\right)^{1/Z_{uid}}$.
- Compute $\overline{P}_{uid} = P_{uid}^{1/Z_{uid}} = g_2^{\delta_{uid}/Z_{uid}}$.
- Set the OSP key related to uid: $DTK_{uid} = \left(\overline{D}_{uid}, \overline{K}_{uid}, \overline{P}_{uid}, S_{uid}\right)$.
- Return the OSP key DTK_{uid} and the recovery key $Z_{uid} \cdot DTK_{uid}$ is transmitted to the OSP, and Us_{uid} ensures the confidentiality of Z_{uid} .

5.1.3. Encryption Phase

1. $\boldsymbol{OSPEncrypt}(GP, PP_{AID}, \mathbb{A}) \rightarrow$ *(IC):* According to the access policy $= (A, \rho)$, where A is a $1 \times \text{n}$ LSSS matrix, the OSP executes this algorithm to generate the intermediate ciphertext IC.

- Define Φ as the set of attribute authorities AAs mentioned in A, and generate E and F:

$$E = \prod_{k\in\Phi} g_1^{\beta_k} = g_1^{\sum_{k\in\Phi}\beta_k}$$

$$F = \prod_{k\in\Phi} e\,(g_1, g_2)^{\alpha_k} = e(g_1, g_2)^{\sum_{k\in\Phi}\alpha_k}$$

- For all $i \in \{1,2,\cdots,1\}$, randomly select $\lambda_i', \tau_i \in Z_r$.
- Generate $\tilde{C}_{1,i} = E^{\lambda_i'}, \tilde{C}_{2,i} = g_2^{\tau_i}$.
- Output the intermediate ciphertext IC:

$$IC = \left(E, F, \left(\tilde{C}_{1,i}, \tilde{C}_{2,i}, \lambda_i', \tau_i\right)_{i\in\{1,2,\cdots,l\}}\right)$$

The OSP sends IC to the DO.

2. $\boldsymbol{DOEncrypt}(GP, IC, \mathbb{A}, m, SKR, PP_{aid}) \rightarrow (CT)$: This algorithm is executed by the DO to encrypt the message M under the access policy $\mathbb{A}$.

- Select a random column vector $v = (s,\ s_2,\ s_3, \cdots s_{\hat{n}}) \in Z_r^{\hat{n}}$, where s is the secret to be shared.
- For all $i \in \{1,2,\cdots,l\}$, generate the share $\lambda_i = A_i \cdot v$.
- Select a random number $\sigma \in Z_r$, then generate the ciphertext CT:

$$CT = \begin{pmatrix} \mathbb{A}, E, C, C_0, \\ C_{1,i}, C_{2,i}, C_{3,i} \end{pmatrix}$$

Among it: $C = M \cdot F^s \cdot CR$, $C_0 = g_2^s$, $C_{1,i} = E^{\sigma} \cdot T_{\rho(i)}^{-\tau_i} \cdot \tilde{C}_{1,i}, C_{2,i} = \tilde{C}_{2,i}, C_{3,i} = (\lambda_i - \lambda_i' - \sigma)$
Subsequently, CT is uploaded to the CSP.

5.1.4. Decryption Phase

1. OSPDecrypt $(GP, CT, DTK_{uid}, u, RL) \rightarrow \left(\overline{CT}\right)$ *:*

- Verify whether the user's key version number matches the ciphertext's version number, that is, determine: $e\left(T_X, \overline{P}_{uid}\right)^{H(x)H(u)} = e\left(\overline{D}_{uid,x},\ g_2\right)$ for each x ∈ S_{uid} . If they match, proceed to the second step; otherwise, abort the decryption.
- If uid ∈ RL, determine whether the attribute set $S'_{uid} = S_{uid} - RL_{uid}$ satisfies the access policy $\mathbb{A}$. If not, stop the decryption. Otherwise, continue the algorithm.
- Define $I = \{i: \rho(i) \in S'_{uid}\}$ as the set of row indices in A marked by the attributes in S_{uid} , and select constants $\omega_i \in Z_r$ such that when the share λ_i is valid for A, $\sum_{i\in I} \lambda_i\, \omega_i = s$.
- Let j = uid, compute the auxiliary decryption element D :

$$D = \frac{e\left(\prod_{k\in\Phi} \overline{K}_{j,k}\,, C_0\right)}{\prod_{i\in I}\left[e\left(C_{1,i} \cdot E^{C_{3,i}}, \overline{P_j}\right) \cdot e\left(\overline{D}_{j,\rho(i)}, C_{2,i}\right)^{1/H(\rho(i))H(u)}\right]^{\omega_i}} = e(g_1, g_2)^{\sum_{k\in\Phi}\alpha_k \cdot s/Z_j}$$

- Set and return the partially decrypted ciphertext $\overline{CT}$: $\overline{CT} = (D, C)$. The OSP sends $\overline{CT}$ to the corresponding EU_{uid} .

2. UserDecrypt $\left(GP, PPR, \overline{CT}, USK_{uid}, z_{uid}\right) \rightarrow (M)$ *:*

- Let j = uid, and compute the revocation recovery component DR :

$$D_R = \frac{e(OCR_1,\ h'_j)}{e(R_j \cdot \prod_{i\in S, i\neq j} h_{n+1-i+j}, OCR_0)} = e(g_1, g_2)^{u\cdot\alpha^{n+1}}$$

- Recover and return M: $M = \frac{C}{D^{Z_j}\cdot D_R} = \frac{M\cdot e(g_1,g_2)^{\prod_{k\in\Phi}\alpha_k\cdot s}\cdot e(g_1,g_2)^{u\cdot\alpha^{n+1}}}{e(g_1,g_2)^{\prod_{k\in\Phi}\alpha_k\cdot s}\cdot e(g_1,g_2)^{u\cdot\alpha^{n+1}}}$

5.1.5. Revocation Phase

1. DORevoke $(GP, SKR, PPR, u_0, RL) \rightarrow (SKR', PPR')$ *:*

- Do add uid or (uid, $\mathrm{RL}_{\mathrm{uid}}$) to RL, where $\mathrm{RL}_{\mathrm{uid}}$ represents the specific attributes of the revoked user.
- Select a random version number $\mathrm{u}_0 \in \mathrm{Z}_\mathrm{r}$, and generate

$$CR' = CR^{\frac{u_0}{u}} = e(g_1, g_2)^{u_0\cdot\alpha^{n+1}}.$$

- Compute $OCR'_0 = g_2^{u_0}, OCR'_1 = (V_R \cdot \prod_{i\in S} h_{n+1-i})^{u_0}$, and $UPK = CR^{\left(\frac{u_0}{u}-1\right)}$.
- Output $\mathrm{CR}', \mathrm{OCR}'_0, \mathrm{OCR}'_1$, UPK.DO replaces OCR_0 and OCR_1 with OCR'_0 and OCR'_1 in PPR, and sends UPK to CSP to update the stored ciphertext.

2. AARevoke $(D_{uid}, RL, u_0, GP) \rightarrow (D'_{udd})$ *:* Each AA receives RL and u_0 , and for the user attributes that are not revoked, updates their corresponding attribute keys: $D'_{uid,x} = PK_{uid}^{t_{\alpha_{aid}}\cdot t_x\cdot H(x)H(u_0)}$. Subsequently, the updated attribute keys are sent to $\mathrm{EU}_{\mathrm{uid}}$ through a secure channel.

3. CTUpdate $(GP, CT, CR') \rightarrow (CT')$ *:*This algorithm is executed by the CSP.

- Generate C': $C' = C \cdot UPK = M \cdot e(g_1,\ g_2)^{\sum_{k\in\Phi}\alpha_k\cdot s} \cdot e(\ g_1,\ g_2)^{u_0\cdot\alpha^{n+1}}$.
- Output $CT' = \left(A, E, C', C_0, \left\{C_{1,i}, C_{2,i}, C_{3,i}\right\}_{\forall i\in\{1,2,\cdots,l\}}\right)$. Finally, the CSP replaces the stored CT with CT' and discards CT.

6. Security Analysis

This section demonstrates the correctness of the outsourced decryption and provides a proof of its IND-CPA security.

6.1. Outsourced Decryption Correctness

$$Let\ D = \frac{DA}{DB}\ .\ Then:$$

$$D_A = e\left(\prod_{k\in\Phi}\overline{K}_{j,k}, C_0\right) = e\left(\prod_{k\in\Phi}\left(g_1^{a_k}\cdot g_1^{\delta_j\cdot\beta_k}\right)^{\frac{1}{Z_j}}, g_2^s\right)$$

$$= e(g_1, g_2)^{\sum_{k\in\Phi}a_k\cdot\frac{s}{Z_j}+\sum_{k\in\Phi}\beta_k\delta_j\cdot\frac{s}{Z_j}} \qquad (1)$$

Let $\mathrm{D}_{\mathrm{B}_1}$ and $\mathrm{D}_{\mathrm{B}_2}$ be the pairings in D_B . We have :

$$D_B = \sum_{i\in I}\left(D_{B_1}\cdot D_{B_2}\right)^{\omega_i}$$

Then*:*

$$D_{B_1} = e\left(C_{1,i}\cdot E^{C_{3,i}}, \overline{P_j}\right)$$

$$= e\left(E^{\sigma} \cdot T_{p(i)}^{-\tau_i} \cdot E^{\lambda_i'} \cdot E^{(\lambda_i - \lambda_i' - \sigma)}, g_2^{\delta_j/Z_j}\right)$$

$$= e\left(g_1^{\sum_{k\in\Phi} \beta_k \lambda_i \cdot \delta_j/Z_j - t_{p(i)} \cdot t_k \cdot \tau_i \cdot \delta_j/Z_j}, g_2\right)$$

$$= e(g_1, g_2)^{\sum_{k\in\Phi} \beta_k \lambda_i \cdot \delta_j/Z_j - t_{p(1)} \cdot t_k \cdot \tau_i \cdot \delta_j/Z_j}$$

$$D_{B_2} = e\left(\overline{D}_{j,\rho(i)}, C_{2,i}\right)^{1/H(\rho(i)) \cdot H(u)}$$

$$= e\left(g_1^{\delta_j t_k \cdot t_{\rho(i)} \cdot H(\rho(i)) \cdot H(u)/Z_j}, g_2^{\tau_i}\right)^{1/H(\rho(i)) \cdot H(u)}$$

$$= e(g_1, g_2)^{\delta_j \cdot t_k \cdot t_{\rho(i)} \cdot \tau_i / Z_j}$$

$$D_B = \sum_{i\in I} \left(e(g_1, g_2)^{\sum_{k\in\Phi} \beta_k \cdot \lambda_i \cdot \delta_j/Z_j}\right)^{\omega_i}$$

$$= e(g_1, g_2)^{\sum_{k\in\Phi} \beta_k \cdot \delta_j/Z_j \cdot \sum_{i\in I} \lambda_i \omega_i}$$

$$= e(g_1, g_2)^{\sum_{k\in\Phi} \beta_k \cdot \delta_j \cdot \frac{s}{Z_j}} \quad (2)$$

Combining (1) and (2), we obtain:

$$D = \frac{D_A}{D_B} = \frac{e(g_1, g_2)^{\sum_{k\in\Phi} a_k \cdot \frac{s}{Z_j} + \sum_{k\in\Phi} \beta_k \delta_j \cdot \frac{s}{Z_j}}}{e(g_1, g_2)^{\sum_{k\in\Phi} \beta_k \cdot \delta_j \cdot \frac{s}{Z_j}}}$$

$$= e(g_1, g_2)^{\sum_{k\in\Phi} \alpha_k \cdot s/Z_j}$$

6.2. User Decryption Correctness

Let $D_R = \frac{DR_1}{DR_2}$ *. Then, we have:*

$$DR_1 = e\left(OCR_1, h_j'\right)$$

$$= e\left(\left(V_R \cdot \prod_{i\in S} h_{n+1-i}\right)^u, g_2^{\alpha^j}\right)$$

$$= e\left(\left(g_1^{\eta} \cdot g_1^{\sum_{i\in S} \alpha^{n+1-i}}\right)^u, g_2^{\alpha^j}\right)$$

$$= e(g_1, g_2)^{(\alpha^j \cdot \eta + \sum_{i\in S} \alpha^{n+1-i+j})u} \quad (3)$$

$$DR_2 = e\left(R_j \prod_{i\in S, i\neq j} h_{n+1-i+j}, OCR_0\right)$$

$$= e\left(g_1^{\alpha^j \cdot \eta} \cdot g_1^{\sum_{i\in S, i\neq j} \alpha^{n+1-i+j}}, g_2^u\right)$$

$$= e(g_1, g_2)^{(\alpha^j \cdot \eta + \sum_{i\in S, i\neq j} \alpha^{n+1-i+j})u} \quad (4)$$

Combining (3) and (4), we obtain:

$$D_R = \frac{DR_1}{DR_2} = e(g_1, g_2)^{u \cdot \alpha^{n+1}}$$

$$M = \frac{C}{D^{Z_j} \cdot D_R} = \frac{M \cdot F^s \cdot CR}{e(g_1, g_2)^{\sum_{k \in \omega} \alpha_k \cdot s} \cdot e(g_1, g_2)^{u \cdot \alpha^{n+1}}} = \frac{M \cdot e(g_1, g_2)^{\sum_{k \in \omega} \alpha_k \cdot s} \cdot e(g_1, g_2)^{u \cdot \alpha^{n+1}}}{e(g_1, g_2)^{\sum_{k \in \omega} \alpha_k \cdot s} \cdot e(g_1, g_2)^{u \cdot \alpha^{n+1}}}$$

6.3. Security Proof

6.3.1. IND-CPA

***Theorem 1*:** If the decisional q-BDHE assumption holds, no polynomial-time adversary can break through our CP-ABE scheme with nonnegligible advantage ε.

Proof:

***Suppose**:* there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can distinguish the valid ciphertext of a random element with a nonnegligible advantage $= Adv(\mathcal{A})$. The simulator $\mathcal{B}$ can solve the decisional DBDH problem with a nonnegligible advantage ε.

Given the parameters of a valid Type-3 pairing $\{G1, G2, GT, g1, g2, r\}$, the challenger c randomly selects elements $a, b, c, z \in Z_r$ and a random bit $K \in \{0,1\}$. If $K = 0$, c sets $Z = e(g_1, g_2)^{abc}$; otherwise, $Z = e(g_1, g_2)^z$. Let $\vec{y} = (g_1, g_2, g_1^a, g_1^b, g_1^c, g_2^a, g_2^b, g_2^c)$. Next, it is assumed that the tuple $\langle \vec{y}, Z \rangle$ is transmitted to $\mathcal{B}$. Then, in the security game described below, $\mathcal{B}$ assumes the role of the challenger c .

***Initialization**:* The adversary $\mathcal{A}$ sends the access policy $\mathbb{A} = (A, \rho)$ to $\mathcal{B}$, and at the same time, S is set as the attribute set mentioned in A. $\mathcal{B}$ records: the set of row indices of matrix A $\{1,2,\dots,l\}$; the set of attribute authorities involved in the policy $A = \{aid \mid \exists i, \rho(i) \in S_{aid}\}$.

***Setup**:* $\mathcal{B}$ selects $t_{aid} = b$, randomly selects $\alpha_{i,0} \in \mathbb{Z}_r (i = 1, \dots, K)$, and calculates $g_2^{\alpha} = g_2^{\sum_{i=1}^{K} \alpha_{i,0}}$; for each $aid \in A$, $\mathcal{B}$ randomly selects $\alpha'_{aid} \in \mathbb{Z}_r$, and implicitly sets the master key component α_{aid} of AA_{aid} *as* $\alpha_{aid} = \frac{ab}{K} + \alpha'_{aid}$; $\mathcal{B}$ randomly selects $\beta_{aid} \in \mathbb{Z}_r$, and calculates PP_{aid} and MSK_{aid} . According to any n>1, run RevSetup (since u is random, u can be set to 1 without affecting the distribution), calculate $h_i = g_1^{c^i}, h'_i = g_2^{c^i}, KR = g_1^{c^{n+1}}$, and generate (PPR, SKR).

***Phase 1**:* The adversary $\mathcal{A}$ requests the key for any attribute set S' , with the constraint that the attribute set S' does not satisfy the access policy $\mathbb{A}$. At the same time, $\mathcal{B}$ selects a user identifier j , randomly selects $\delta_{uid} \in \mathbb{Z}_r$ to generate (UPK_{uid}, USK_{uid}) ; for each $aid \in A$, calculate $K_{uid,aid} = g_1^{\alpha_{aid} + \delta_{uid} \cdot \beta_{aid}} = g_1^{\frac{ab}{K} + \alpha'_{aid} + \delta_{uid} \cdot \beta_{aid}}$; for $x \in S' \cap S_{aid}$, calculate $D_{uid,x} = g_1^{\delta_{uid} \cdot t_{aid} \cdot t_x \cdot H(x) \cdot H(u)} = g_1^{\delta_{uid} \cdot b \cdot t_x \cdot H(x) \cdot H(u)}$. Send $UPK_{uid}, USK_{uid}, K_{uid} = \{K_{uid,aid}\}, D_{uid} = \{D_{uid,x}\}$ *to* $\mathcal{A}$. $\mathcal{A}$ can execute UserKeyGen_{out} to generate the proxy decryption key DFK_{uid} and the recovery key z_{uid} .

Challenge: The adversary $\mathcal{A}$ generates two messages of equal length m_0, $m_1 \in GT$ for the challenge. At this time, $\mathcal{B}$ simulates the execution of OSPEncrypt and DoEncrypt. $\mathcal{B}$ selects a random $t \in \{0,1\}$, then calculates $E = \prod_{aid \in A} g_1^{\beta_{aid}}$ and $F = \prod_{aid \in A} e(g_1, g_2)^{\alpha_k} = e(g_1, g_2)^{\sum_{aid \in A} \left(\frac{ab}{K} + \alpha'_k\right)} = e(g_1, g_2)^{ab \ \sum \alpha'_k}$. Using c as the secret value s, construct the column vector $v = (c, s_2, \dots, s_n)$, and calculate the share $\lambda_i = A_i \cdot v$. Then, for $i = \{1, \dots, l\}$, $\mathcal{B}$ randomly selects $\sigma, \tau_i \in \mathbb{Z}_r$, and sets: $C = m_t \cdot F^s \cdot CR = m_t \cdot e(g_1, g_2)^{(ab \ \sum \alpha'_k) \cdot c} \cdot e(g_1, g_2)^{c^{n+1}} = m_t \cdot Z \cdot e(g_1, g_2)^{\sum \alpha'_k \cdot c + c^{n+1}}, C_0 = g_2^c, C_{1,i} = E^{\sigma} \cdot T_{\rho(i)}^{-\tau_i} \cdot E^{\lambda'_i}, C_{2,i} = g_2^{\tau_i} \cdot C_{3,i} = \lambda_i - \lambda'_i - \sigma (\lambda'_i \in \mathbb{Z}_r)$. Finally, $\mathcal{B}$ defines the challenge ciphertext $CT^* = \left(\mathbb{A}, E, C, C_0, \{C_{1,i}, C_{2,i}, C_{3,i}\}\right.$ and sends it to $\mathcal{A}$.

Phase 2: Identical to Phase 1, with the only restriction that the ciphertext to be decrypted cannot be CT^*.

Guess: Finally, $\mathcal{A}$ returns a value $t' \in \{0,1\}$, which represents the guess for t . If $t' = t$, $\mathcal{B}$ outputs $K = 0$, indicating that $Z = e(g_1, g_2)^{abc}$, which means that according to the access policy A, CT has been constructed as a valid encryption of m_t . If $t' \neq t$, $\mathcal{B}$ outputs $K = 1$, indicating that $Z = e(g_1, g_2)^z$, which means that $\mathcal{A}$ cannot obtain any information about m_t . Consider the following cases:

- If $Z = e(g_1, g_2)^{abc}$, $\mathcal{B}$ has the same advantage as $\mathcal{A}$ in winning the game. Thus, $\Pr\left[B\left(\vec{y}, Z = e(g_1, g_2)^{abc}\right) = 0\right] = 1/2 + \varepsilon$.

- If $Z = e(g_1, g_2)^z$, then the message m_v is randomly masked, and the advantage of $\mathcal{B}$ in this case is: $\Pr[B(\vec{y}, Z = e(g_1, g_2)^z) = 0] = 1/2$.

Considering the above cases, the advantage of the simulator $\mathcal{B}$ in solving the DBDH problem is: $\mathrm{Adv}_{\mathrm{DBDH}}(\mathcal{B}) = \left|\frac{1}{2} \cdot \left(\frac{1}{2} + \varepsilon\right) + \frac{1}{2} \cdot \frac{1}{2} - \frac{1}{2}\right| = \frac{\varepsilon}{2}$. Since ε is assumed to be nonnegligible, it can be seen that $\mathrm{Adv}_{\mathrm{DBDH}}(\mathcal{B}) = \frac{\varepsilon}{2}$ is also nonnegligible, which contradicts the DBDH hardness assumption. Therefore, no PPT adversary $\mathcal{A}$ can break the CPA security of the proposed scheme.

6.3.2. Forward Security

In this scheme, for a user to correctly decrypt the ciphertext, they must first go through the outsourcing server. The outsourcing server uses the outsourcing decryption key to pre-decrypt the ciphertext to obtain the pre-decrypted ciphertext, and sends the pre-decrypted ciphertext and the original text component C to the user. The user then uses the recovery key they hold to perform the final decryption on the pre-decrypted ciphertext to obtain the plaintext. When some attributes of the user are revoked, DO updates the revocation component and the version number. When the revoked user requests decryption again, the calculated revocation recovery component D_R is associated with the new version number, and the user cannot use the saved old pre-decrypted ciphertext to decrypt and obtain the plaintext. When a user is revoked, the proxy server will directly delete the user's outsourcing decryption key and no longer respond to the user's requests. Therefore, the scheme has forward security.

6.3.3. Resistance to Collusion Attacks

This scheme has the ability to resist collusion attacks to ensure data security. When a revoked user colludes with a legitimate user, due to the scheme, the private keys assigned to users during registration are different, and the attribute keys generated for each user are also related to the user identifier. Therefore, even if multiple users collude, the attribute keys they generate are independent, and they cannot jointly decrypt the file.

On the other hand, when a revoked user colludes with the proxy, even if the malicious proxy allows the revoked user to pass the first two steps of the OSPDecrypt process, the equation : $D_{B_2} = e\left(\overline{D}_{j,\rho(i)}, C_{2,i}\right)^{1/H(\rho(i))\cdot H(u)} = e\left(g_1^{\delta_j t_k \cdot t_{\rho(i)} \cdot H(\rho(i)) \cdot H(u)/Z_j}, g_2^{\tau_i}\right)^{1/H(\rho(i))\cdot H(u_0)}, H(\rho(i)) \cdot H(u_0)$cannot be correctly canceled during the outsourcing decryption calculation to obtain D. This leads to an incorrect calculation of D, ultimately preventing the revoked user and the malicious proxy from jointly decrypting the file.

7. Performance Analysis

This section compares proposed scheme with Liu89[15], Yin[7], Li[11], and Luo[19] through theoretical analysis.

7.1. Theoretical Analysis

To begin, we compare the functional features of these schemes, focusing on aspects such as attribute revocation and outsourced decryption. The results are summarized in Table 2.

As shown, Li [19] supports neither attribute nor user revocation, nor does it provide outsourced decryption. Luo [20] supports outsourced encryption, outsourced decryption, and attribute revocation, but it adopts a single-authority model, which brings the key escrow problem. Yin [7]supports user revocation but lacks outsourced encryption or decryption. Both Liu [15] and the proposed scheme use a multi-authority architecture and support both attribute and user revocation; however, Liu [15]does not support outsourced encryption.

In comparison, the proposed scheme not only achieves outsourced encryption and decryption and supports attribute and user revocation but also employs a multi-authority framework to enhance security.

Table 2. Comparison of features of previous schemes and the proposed scheme

Schemes	Auths	Outsourced Enc	Outsourced Dec	Revocation
Liu[19]	Multi- Authority	No	Yes	Attribuate/User
Yin[7]	Single Authority	No	No	User
Li[15]	Multi- Authority	Yes	No	No
Luo[11]	Single Authority	Yes	Yes	Attribuate/User
Ours	Multi- Authority	Yes	Yes	Attribuate/User

Secondly, analyzing these schemes theoretically in terms of user key size, ciphertext size, and storage cost of public parameters. In the comparison, the following notations are used: $|\mathbb{Z}_r|, |\mathbb{G}|, |\mathbb{G}_1|, |\mathbb{G}_2|, |\mathbb{G}_T|$ denote the element sizes in the groups $\mathbb{Z}_r, \mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, respectively; N_{aa} denotes the number of attributes for a user; N_a denotes the number of attributes contained in the access structure; N_{ua} denotes the number of user attributes; s_A denotes the size of the access policy A; r denotes the number of users in the revocation list; $N_{aa,A}$ denotes the number of attribute authorities. We assume all groups are of the same prime order, so the element sizes in different groups are identical.

Table 3. Comparison of storage overhead

Schemes	User Decryption Key	Ciphertext
Liu[19]	$\|\mathbb{Z}_r\|$	$2\|\mathbb{G}_1\| + 2\|\mathbb{G}_T\| + \|\mathbb{Z}_r\|$
Yin[7]	$(5+2r)\|\mathbb{G}\| + \|\mathbb{Z}_r\|$	$\|\mathbb{G}_T\| + (6+2s_A+2r)\|\mathbb{G}\|$
Li[15]	$(2N_{ua}+3)\|\mathbb{G}\|$	$(3s_A+1)\|\mathbb{G}\| + \|\mathbb{G}_T\|$
Luo[11]	$(N_{aa}+3)\|\mathbb{G}\|$	$(N_{ua}+N_a+2)\|\mathbb{G}\| + \|\mathbb{G}_T\|$
Ours	$\|\mathbb{Z}_r\|$	$S_A\|\mathbb{Z}_r\|+(1+S_A)\|\mathbb{G}_1\| + (1+S_A)\|\mathbb{G}_2\| + \|\mathbb{G}_T\|$

The results are shown in

Table 3. It can be seen that the user decryption key size in Liu [19] is smaller than that in the other three schemes and is equal to the key size in the proposed scheme. Liu [19] has the smallest ciphertext size, while the proposed scheme outperforms Luo[11] and Yin [7]in terms of ciphertext size, with Yin[7]'s ciphertext size being nearly double that of the proposed scheme. The ciphertext size of Li[15] is comparable to that of the proposed scheme.

Finally, theoretically analyzing the computational costs of the proposed scheme and comparing the schemes during the encryption, decryption, and revocation phases. To ensure objectivity and accuracy, the theoretical comparisons for specific cases are expressed using the parameter symbols from the original schemes. Here, γ denotes the number of attributes shared between the access matrix structure and the user attribute set; $T_{\mathbb{G}}^m, T_{\mathbb{G}_1}^m, T_{\mathbb{G}_2}^m, T_{\mathbb{G}_T}^m$ denote multiplication operations in groups $\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, respectively; $T_{\mathbb{G}}^e, T_{\mathbb{G}_1}^e, T_{\mathbb{G}_2}^e, T_{\mathbb{G}_T}^e$ denote exponentiation operations in the respective groups; p denotes one bilinear pairing operation. In Yin[7], k represents the minimum number of attributes required to satisfy the access policy.

Table 4. Comparison of computation overhead

Schemes	DO encryption	US decryption
Liu[19]	$(N_{aa,A}+1)T_{\mathbb{G}_T}^m + N_a T_{\mathbb{G}_1}^m + p + 2T_{\mathbb{G}_1}^e$	$T_{\mathbb{G}_T}^m + T_{\mathbb{G}_1}^e$
Yin[7]	$T_{\mathbb{G}_T}^e + (3+2s_A)T_{\mathbb{G}}^e + (s_A+r)T_{\mathbb{G}}^m$	$(5+k+2r)p + (k+1)T_{\mathbb{G}_T}^e + (r+1)T_{\mathbb{G}}^e$
Li[15]	$(2s_A+1)T_{\mathbb{G}}^m + 2s_A T_{\mathbb{G}}^e$	$T_{\mathbb{G}}^e + T_{\mathbb{G}}^m$
Luo[11]	$(N_{ua}+N_a+2)T_{\mathbb{G}}^e + T_{\mathbb{G}_T}^e$	$(3\gamma+1)P + T_{\mathbb{G}_T}^e$
Ours	$2S_{\mathbb{A}}T_{\mathbb{G}_1}^m + (1+S_{\mathbb{A}})T_{\mathbb{G}_1}^e + T_{\mathbb{G}_2}^e + 2T_{\mathbb{G}_T}^m + T_{\mathbb{G}_T}^e$	$3T_{\mathbb{G}_T}^m + T_{\mathbb{G}_T}^e$

The experimental results are shown in

Table 4. When the number of attributes in the access structure is the same, the proposed scheme outperforms Yin[7] and Liu[19] in encryption computational cost. Furthermore, similar to Li[15] and Liu [15], the proposed scheme supports an outsourced decryption mechanism, requiring users to perform only a few exponentiation and multiplication operations in the cyclic group GT to complete decryption, which is more efficient than Luo[11] and Yin[7]. The comparison shows that the overall computational overhead of the proposed scheme is lower than that of existing schemes, making it more suitable for deployment in resource-constrained lightweight terminal devices.

7.2. Experimental Evaluation

To further analyze the performance of this scheme, the proposed scheme was implemented using the JPBC library in the environment of the IntelliJ IDEA compiler. The operating system is 64-bit Windows 10, the processor is AMD Ryzen 74800H with Radeon Graphics @ 2.90GHz, and the memory is 16GB. The number of attributes in the access policy increases from 1 to 100, and the experiment was conducted 10 times. The average value of the 10 experimental results was taken as the final result of this experiment to ensure the accuracy of the experiment.

Figure 3 shows the running time of the main algorithms of the proposed scheme under different numbers of user attributes. It can be seen from the figure that the time for key generation, outsourced encryption, and data owner encryption increases linearly with the growth of the number of user attributes, while the user decryption time remains basically stable and is relatively short. The time consumption of the attribute revocation algorithm also increases roughly linearly with the increase in the number of attributes.

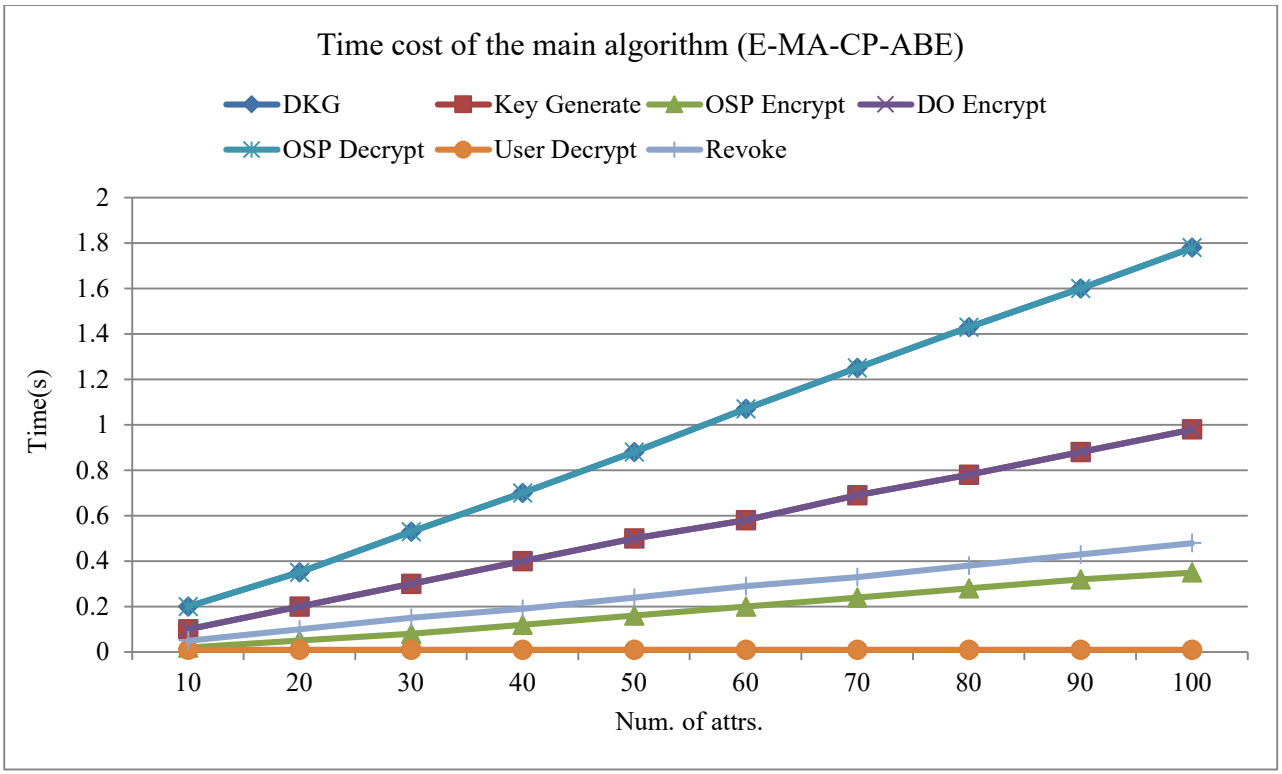

Fig. 3 The running time of the main algorithms

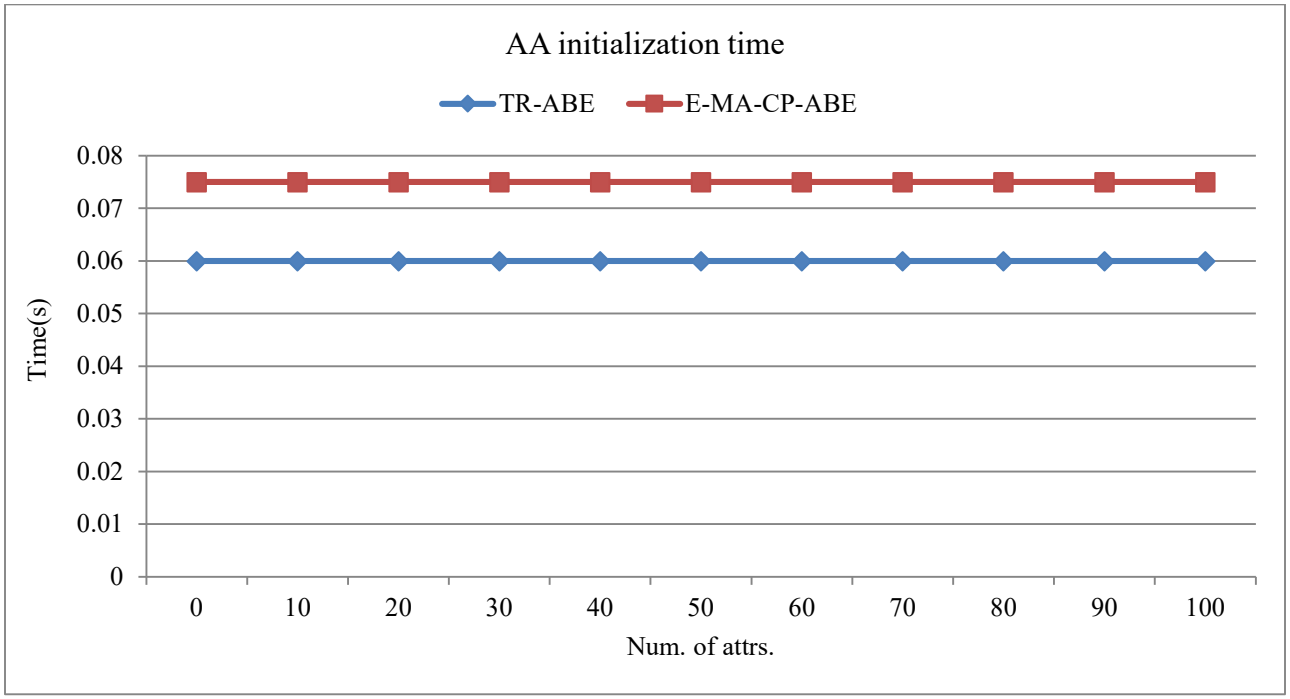

Fig. 4 The running time of AA initialization

As shown in Figure 4, the system initialization time of this scheme is close to that of the scheme[11]. Although the initialization time of this scheme is slightly higher than that of the scheme[11], the initialization parameters of this scheme are issued by the trusted DO, so it has higher security.

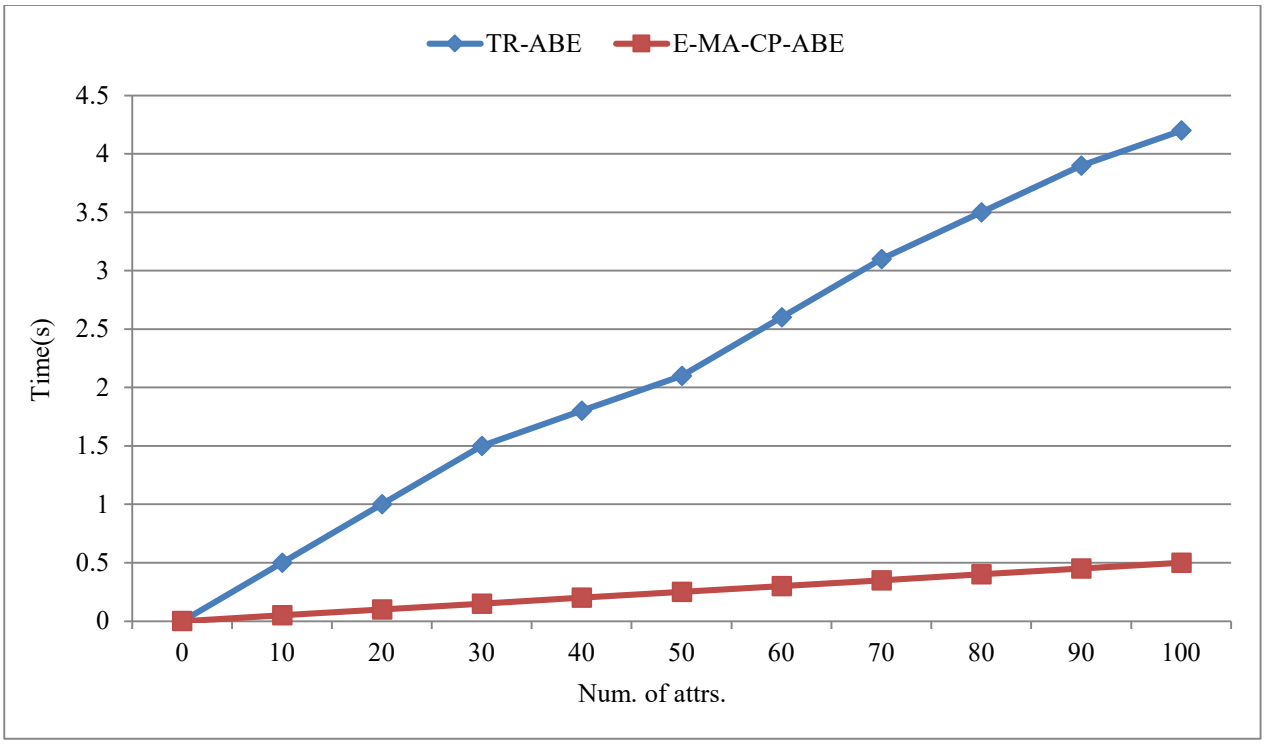

Fig. 5 The running time of key generation

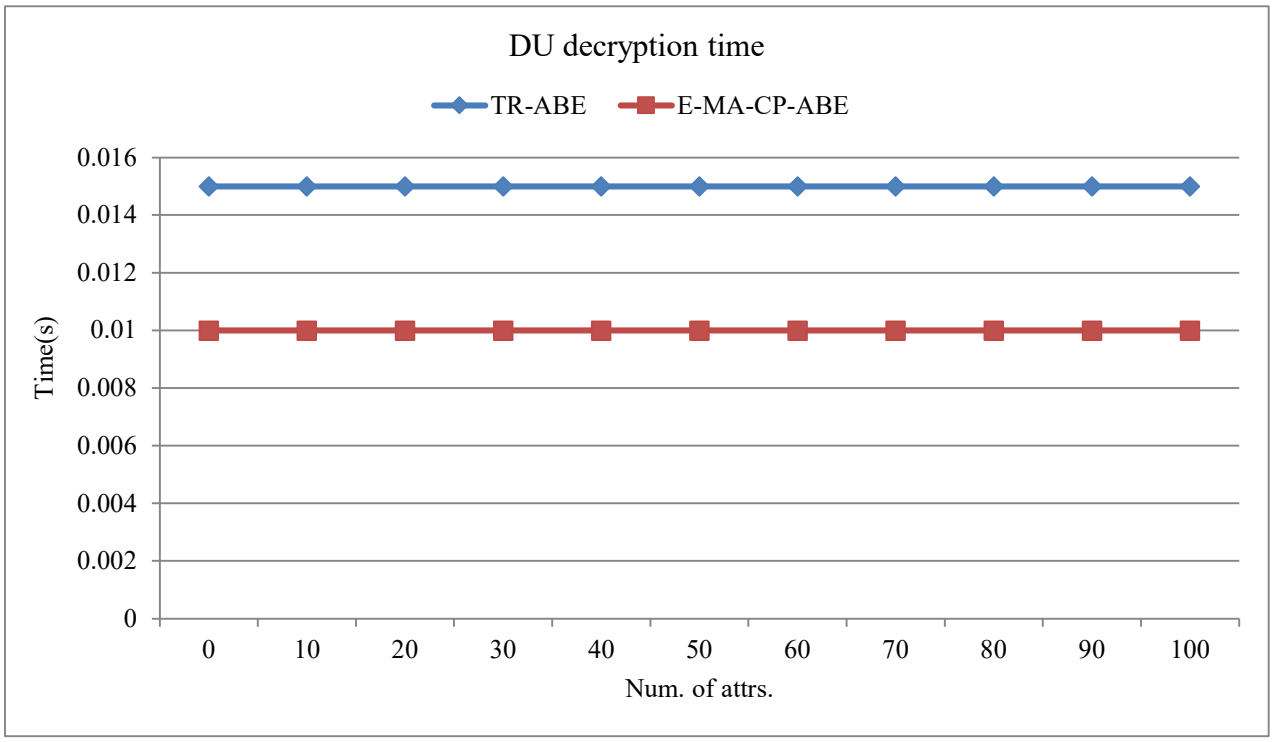

Fig. 6 The running time of decryption

In Figure 5, the key generation time of both schemes is roughly linear with the number of user attributes, but the slope of this scheme is smaller, which improves the performance to a certain extent.

Figure 6 illustrates that most of the ciphertext decryption calculations of both schemes are outsourced to the cloud server, and local users only need to perform simple calculations, so the decryption time is at a constant level. The user decryption time of this scheme is very close to that of the scheme [11], and is only slightly lower than the latter.

8. Conclusion

This paper proposes and implements a multi-authority attribute-based encryption scheme that supports outsourced decryption and dynamic revocation. Instead of relying on a single trusted entity, the system master key is securely distributed among multiple attribute authorities using a distributed key generation protocol, which effectively eliminates single points of failure. User attributes are managed collectively by these authorities—this not only reduces the workload on any individual authority but also strengthens the overall security of the system. On the efficiency side, computationally heavy encryption and decryption operations are outsourced to external servers, while user key privacy is preserved through the use of randomization factors. Dynamic revocation is achieved by introducing version numbers that link both ciphertexts and user keys to specific versions. This means that when a revocation occurs, the keys of unaffected users and the corresponding ciphertexts need to be updated in a timely manner.

Security analysis shows that the scheme achieves selective security in the standard model and is resistant to collusion attacks as well as chosen-plaintext attacks. Performance evaluation indicates that, while maintaining the same level of security, the proposed scheme significantly reduces the computational burden on the user side compared to existing approaches, making it well-suited for resource-constrained mobile environments. In addition, it outperforms benchmark multi-authority schemes in terms of key generation, encryption, and decryption performance.

Future Work

In practice, deploying a fully distributed key generation protocol along with outsourcing servers inevitably adds extra rounds of interaction, which results in higher communication overhead during decryption compared to other schemes. Finding ways to simplify or reduce these interactions—so as to lower the overall communication cost—remains an important direction for future work.

Moreover, the correctness of the revocation mechanism currently assumes perfect synchronization. In real-world deployments, issues like version rollback and update latency would need to be addressed through further refinements. The scheme could also be extended to achieve security against chosen-ciphertext attacks.

Funding Statement

This work was supported by the National College Student Innovation and Entrepreneurship Training Program (Grant No. 202410288089Z)

Acknowledgments

The proposed algorithm was designed and implemented by Bin Ge. Yi Chen and Chanyi Gong drafted the full manuscript. Zhaojie Bu and Jinyan Cui contributed to the review and refinement of the detailed content of the manuscript. Chungen Xu and Gangqiang Duan supervised this study and verified the feasibility of the implementation. They also reviewed the manuscript and provided revision suggestions. In addition, Chungen Xu offered institutional support.

References

[1] Amit Sahai, and Brent Waters, "Fuzzy Identity-Based Encryption," *Advances in Cryptology – EUROCRYPT*, vol. 3494, pp. 457-473, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[2] John Bethencourt, Amit Sahai, and Brent Waters, "Ciphertext-Policy Attribute-Based Encryption," *IEEE Symposium on Security and Privacy*, pp. 321-334, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[3] Junbeom Hur, and Dong Kun Noh, "Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214-1221, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[4] Vipul Goyal et al., "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89-98, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[5] Melissa Chase, "Multi-Authority Attribute Based Encryption," *Theory of Cryptography*, vol. 4392, pp. 515-534, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[6] Van-Hoan Hoang, Elyes Lehtihet, and Yacine Ghamri-Doudane, "Forward-Secure Data Outsourcing Based on Revocable Attribute-Based Encryption," *15th International Wireless Communications & Mobile Computing Conference*, pp. 1839-1846, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[7] Hongjian Yin et al., "A Traceable CP-ABE Scheme Supporting Dynamic Revocation and Efficient Decryption for Medical Data Sharing," *IEEE Internet of Things Journal*, vol. 12, no. 24, pp. 53610-53622, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[8] Shuaishuai Chang et al., "T-ABE: A Practical ABE Scheme to Provide Trustworthy Key Hosting on Untrustworthy Cloud," *IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 960-967, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Sravya Gudipati, Syam Kumar Pasupuleti, and R. Padmavathy, "Quantum-Resistant Traceable, Revocable, and Key Escrow-Free CP-ABE for Cloud Storage," *Peer-to-Peer Networking and Applications*, vol. 18, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[10] Alejandro Peñuelas-Angul, Claudia Feregrino-Uribe, and Miguel Morales-Sandoval, "A Revocable Multi-Authority Attribute-Based Encryption Scheme for Fog-Enabled IoT," *Journal of Systems Architecture*, vol. 155, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[11] L.F. Guo, X.M. Xing, and H. Guo, "An Efficient Traceable and Revocable Attribute-Based Encryption Scheme for Cloud Storage," Journal of Cryptology, vol. 10, no. 1, pp. 131-145, 2023. [Google Scholar]

[12] Kai Zhang et al., "A Traceable and Revocable Multiauthority Attribute-Based Encryption Scheme with Fast Access," *Security and Communication Networks*, vol. 2020, pp. 1-14, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Melissa Chase, and Sherman S.M. Chow, "Improving Privacy and Security in Multi-Authority Attribute-Based Encryption," *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pp. 121-130, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[14] Huang Lin et al., "Secure Threshold Multi Authority Attribute Based Encryption without a Central Authority," *Information Sciences*, vol. 180, no. 13, pp. 2618-2632, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[15] Y. Liu, S. Xu, and Z. Yue, "Partially Hidden Policy Multi-Authority CP-ABE Scheme with Constant Length Ciphertext", Journal on Communications, vol. 45, no. 8, pp. 20-36, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Guangli Xiang et al., "An Attribute Revocable CP-ABE Scheme," *Seventh International Conference on Advanced Cloud and Big Data*, pp. 198-203, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] Yousheng Zhou et al., "TRE-DSP: A Traceable and Revocable CP-ABE based Data Sharing Scheme for IoV with Partially Hidden Policy," *Digital Communications and Networks*, vol. 11, no. 2, pp. 455-464, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[18] Shahzad Khan et al., "OO-ABMS: Online/Offline-Aided Attribute-Based Multi-Keyword Search," *IEEE Access*, vol. 9, pp. 114392-114406, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[19] Juyan Li et al., "Online/Offline MA-CP-ABE with Cryptographic Reverse Firewalls for IoT," *Entropy*, vol. 25, no. 4, pp. 616, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Wei Luo et al., "FOC-PH-CP-ABE: An Efficient CP-ABE Scheme with Fully Outsourced Computation and Policy Hidden in the Industrial Internet of Things," *IEEE Sensors Journal*, vol. 24, no. 18, pp. 28971-28981, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Brent Waters, "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization," *International Workshop on Public Key Cryptography*, vol. 6571, pp. 53-70, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[22] Kwangsu Lee, "Ciphertext Outdate Attacks on the Revocable Attribute-Based Encryption Scheme with Time Encodings," *IEEE Access*, vol. 7, pp. 165122-165126, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[23] Yuhan Bai et al., "CR-FH-CPABE: Secure File Hierarchy Attribute-Based Encryption Scheme Supporting User Collusion Resistance in Cloud Computing," *IEEE Internet of Things Journal*, vol. 11, no. 10, pp. 17727-17739, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[24] Sangjukta Das, and Suyel Namasudra, "Multiauthority CP-ABE-Based Access Control Model for IoT-Enabled Healthcare Infrastructure," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 821-829, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Zhongxiang He et al., "Revocable and Traceable Undeniable Attribute-Based Encryption in Cloud-Enabled E-health Systems," *Entropy*, vol. 26, no. 1, pp. 1-17, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[26] Guojun Wang, Qin Liu, and Jie Wu, "Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services," *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pp. 735-737, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[27] Wu Zhaoxia, and Jiang Xu, "Multi-Authority Attribute-Based Security Solution for Policy Renewability in Cloud Healthcare Environments," *Application Research of Computers*, vol. 42, no. 6, pp. 1868-1872, 2025. [CrossRef] [Publisher Link]

[28] Chunpeng Ge et al., "Attribute-Based Proxy Re-Encryption With Direct Revocation Mechanism for Data Sharing in Clouds," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 2, pp. 949-960, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[29] Jiguo Li et al., "PH-MG-ABE: A Flexible Policy-Hidden Multigroup Attribute-Based Encryption Scheme for Secure Cloud Storage," *IEEE Internet of Things Journal*, vol. 12, no. 2, pp. 2146-2157, 2025. [CrossRef] [Google Scholar] [Publisher Link]