sensors

MDPI

Article

An Efficient Pairing-Free Ciphertext-Policy Attribute-Based Encryption Scheme for Internet of Things

Chong Guo [1], Bei Gong [1], Muhammad Waqas [2,3,*], Hisham Alasmary [4], Shanshan Tu [1] and Sheng Chen [5]

[1] College of Computer Science, Beijing University of Technology, Beijing 100124, China; chongguo@emails.bjut.edu.cn (C.G.); gongbei@bjut.edu.cn (B.G.); sstu@bjut.edu.cn (S.T.)
[2] School of Computing and Mathematical Sciences, Faculty of Engineering and Science, University of Greenwich, London SE10 9LS, UK; engr.waqas2079@gmail.com
[3] School of Engineering, Edith Cowan University, Joondalup, WA 6027, Australia
[4] Department of Computer Science, King Khalid University, Abha 62529, Saudi Arabia; alasmary@kku.edu.sa
[5] School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK; sqc@ecs.soton.ac.uk
* Correspondence: engr.waqas2079@gmail.com

Abstract: The Internet of Things (IoT) is a heterogeneous network composed of numerous dynamically connected devices. While it brings convenience, the IoT also faces serious challenges in data security. Ciphertext-policy attribute-based encryption (CP-ABE) is a promising cryptography method that supports fine-grained access control, offering a solution to the IoT's security issues. However, existing CP-ABE schemes are inefficient and unsuitable for IoT devices with limited computing resources. To address this problem, this paper proposes an efficient pairing-free CP-ABE scheme for the IoT. The scheme is based on lightweight elliptic curve scalar multiplication and supports multi-authority and verifiable outsourced decryption. The proposed scheme satisfies indistinguishability against chosen-plaintext attacks (CPA) under the elliptic curve decisional Diffie–Hellman (ECDDH) problem. Performance analysis shows that our proposed scheme is more efficient and better suited to the IoT environment compared to existing schemes.

Keywords: ciphertext-policy attribute-based encryption; pairing-free; access control; Internet of Things

Citation: Guo, C.; Gong, B.; Waqas, M.; Alasmary, H.; Tu, S.; Chen, S. An Efficient Pairing-Free Ciphertext-Policy Attribute-Based Encryption Scheme for Internet of Things. *Sensors* **2024**, *1*, 0. https://doi.org/

Academic Editor: Raffaele Bruno

Received: 10 September 2024
Revised: 1 October 2024
Accepted: 21 October 2024
Published:

1. Introduction

The development of the Internet of Things (IoT) and the advancement of 5G technology have greatly reduced the deployment costs of IoT devices and facilitated the application and expansion of the IoT [1,2]. As the number of IoT devices continues to grow and application scenarios become more complex and diverse, the volume of data generated in the IoT has surged. However, most current IoT devices are equipped with low-voltage and low-power processors, small storage space, and relatively low energy storage. This makes efficient data sharing and low-energy-consumption data processing critical in IoT [3,4]. Reducing the computational costs of a device is key to lowering its energy consumption, improving operational stability, and extending its lifecycle [5,6].

As an important innovation in the development of the Internet, cloud computing technology is currently very mature and widely used. Cloud servers provide abundant computing resources and massive storage space, allowing IoT devices to offload complex computational tasks to the cloud and store data for extended periods [7]. With the help of cloud servers, data can be conveniently shared between devices [8,9]. Utilizing cloud computing to perform specific computing tasks in the IoT can improve the speed of data processing, reduce computational costs, and extend the lifecycle of devices [10]. However, cloud service providers (CSPs) are untrusted third-party organizations, and the cloud computing services and cloud storage services they provide are uncontrollable for data

owners [11–14]. To eliminate the security risks associated with leakage, data should be encrypted before being uploaded to hide sensitive information [15]. Data sharing in the IoT involves a many-to-many relationship. From a data perspective, sharing one piece of data is a one-to-many relationship. However, traditional cryptographic schemes, such as advanced encryption standard (AES) and Rivest–Shamir–Adleman (RSA) are only suitable for one-to-one ciphertext sharing, requiring data to be encrypted multiple times for different recipients. This results in an increased number of encryption tasks, leading to higher computational costs and energy consumption for IoT devices. Furthermore, storing these multiple ciphertexts in the cloud can significantly increase the required storage space.

Attribute-based encryption (ABE) schemes can effectively solve the above problem [16–18]. They can set up an attribute-based access structure by defining a set of attributes which can describe the identity of legitimate users under the premise that the data visitors are unknown [19–21]. The user can successfully decrypt the ciphertext only when the attributes satisfy the access structure. Ciphertext-policy ABE (CP-ABE) was proposed by Bethencourt et al. [22] based on the ABE of [23]. CP-ABE fuzzes users with attributes and achieves fine-grained access control with the access structure, making it suitable for one-to-many data sharing relationships between huge numbers of devices in the IoT [24]. In CP-ABE, the ciphertext is associated with an access policy and the key is generated based on a set of attributes. This allows the data owner to establish the access policy autonomously and share the data with various users while safeguarding the access control of the data [16,25]. Currently, most CP-ABE schemes use bilinear pairing as the primary operation [26]. However, bilinear pairing is a complex operation that is considered to have the highest computational cost in pairing-based cryptographic protocols [27]. Therefore, CP-ABE schemes based on bilinear pairing suffer from long encryption times and high-energy consumption. These limitations render their direct application a challenging task in IoT environments, where computational resources are scarce and energy storage is limited.

To address these challenges, the scheme [28] keeps the length of the ciphertext constant and independent of the number of attributes chosen, thus reducing the need for bilinear pairings during decryption. In addition, outsourcing complex decryption computations to cloud servers can further reduce the computational burden on devices [27,29]. In recent years, several elliptic curve cryptography (ECC)-based ABE schemes without pairing have been proposed [30–35]. On the same elliptic curve, the computational cost of a single bilinear pairing is more than twice that of the scalar multiplication method, while the computational cost of an asymmetric bilinear pairing is 10–40 times that of the scalar multiplication method. Replacing bilinear pairing with scalar multiplication as the main operation of CP-ABE can dramatically reduce the overall computational cost of the system, which provides a novel and feasible approach for the application of CP-ABE in IoT. Against this background, this paper proposes an efficient data-sharing scheme based on pairing-free CP-ABE for IoT. The main contributions of this work are summarized as follows.

(1) We propose a pairing-free CP-ABE scheme for the IoT that uses elliptic curve scalar multiplication as the primary operation. This design retains the fine-grained access control features of CP-ABE while significantly reducing computational complexity, making it more suitable for IoT devices with limited computing resources.
(2) Our scheme establishes multiple attribute authorities (AAs) to decentralize the attribute management. In this way, our scheme can avoid the system bottleneck and key escrow problems caused by a single AA in traditional CP-ABE schemes, and guarantee rapid response to requests from a massive amount of IoT devices.
(3) We ensure data security with the hybrid cryptographic method, encrypting data with symmetric cryptographic algorithms and encrypting keys with CP-ABE. Moreover, linear secret sharing scheme (LSSS) is adopted to enhance the expression of the access policy and to provide flexible access control.
(4) Our scheme supports verifiable outsourced decryption. With outsourced computing, only a small amount of computation is required to decrypt the ciphertext on the IoT

device, which effectively reduces the decryption cost. Before decrypting with the symmetric key, the device can determine whether the data has been tampered through the integrity verification function.

(5) We conducted a detailed security analysis and performance analysis of the proposed scheme and the results prove that it is both secure and efficient.

The remainder of this paper is organized as follows. Section 2 introduces the related work on ABE, and some preliminaries are given in Section 3. Section 4 describes the system model and the security model of our scheme, while an efficient pairing-free CP-ABE scheme for the IoT is proposed in Section 5. Sections 6 and 7 are devoted to the security analysis and performance analysis of our scheme, respectively. Our conclusions are presented in Section 8.

2. Related Work

2.1. Typical ABE Schemes

Amit and Waters [23] extended ABE by abstracting the features of a user's identity based on the identity-based encryption (IBE) [36]. By defining specific identities through a set of attributes, ABE overcomes the limitations of IBE that use a single identification information to determine users' identities, which enables fine-grained access control for users while still providing privacy protection for them. Although the work [23] successfully introduced the concept of attributes, access control was implemented by gate access structure, which suffers from inefficiency as well as limitations. Goyal et al. [37] first proposed key-policy ABE (KP-ABE). KP-ABE embeds an access policy in the key, and correlates the set of attributes with the ciphertext, so that the ciphertext can be decrypted only if the set of attributes satisfies the access policy. To make the expression of the access structure more flexible, the work [37] used a monotonic access tree.

Bethencourt et al. [22] proposed CP-ABE with the opposite structure to KP-ABE. Compared to KP-ABE, CP-ABE matches ciphertexts with keys and better reflects the concept of abstracting users into roles. This makes it easier to provide flexible and fine-grained control over user access to data. However, CP-ABE of [22] also uses a monotonic access tree. Ibraimi et al. [38] proposed a CP-ABE scheme that supports access structures represented by Boolean operator formulas. Although this scheme has low computational efficiency, it successfully removes the restriction where only the access tree can be used. Waters [39] proposed an LSSS and designed a CP-ABE scheme with LSSS. Compared to the previous schemes, the scheme of [39] provides more fine-grained access control to users, but only proves the security under the decisional parallel bilinear Diffie–Hellman exponent assumption.

Nishide et al. [40] first implemented hiding the access policy, making it impossible for any user to obtain any information about the access policy associated with the encrypted data from the ciphertext, and proved the security of the scheme under the decisional bilinear Diffie–Hellman (DBDH) and discrete logarithm (DL) assumptions. Lewko and Waters [19] proposed a multi-authority ABE system that diminishes the reliance on the central authority. More specifically, in the scheme of [19], after the initial parameters are created, any party can be an authority and users can encrypt data based on any attributes issued by any authority. Zhang et al. [41] proposed a large universe multi-authority CP-ABE with white-box traceability in the prime order groups, which removes the limitation where traceable multi-authority CP-ABE cannot support large universe and achieves effective tracking of users who maliciously compromise keys.

All the aforementioned schemes are typical ABE schemes. Although they bear certain shortcomings, these ABE schemes open up a range of viable research directions and provide a foundation for subsequent research.

2.2. ABE Schemes with Outsourced Computation

With the continuous breakthroughs in cloud computing technology, leveraging cloud services to share the computational load for resource-constrained devices has emerged as a

new research trend. It is essential to safeguard sensitive information when uploading data to cloud servers, since CSPs are not entirely trustworthy. Green et al. [42] first proposed an ABE scheme with outsourced decryption. More specifically, in the scheme of [42], for any ABE ciphertext satisfied by the set of user attributes, the cloud server converts these ciphertexts into constant size ElGamal ciphertexts using the transformation key without access to the contents of any ciphertext, and then sends them to the user. This scheme significantly reduces the computational cost on the user and ensures the confidentiality of the data. However, the scheme has a drawback: it cannot verify the integrity and correctness of the ElGamal ciphertexts. To address this issue, Lai et al. [43] refined the outsourced decryption verification function to improve the reliability of their scheme, at the cost of increased computation and communication. Lin et al. [44] constructed an ABE with verifiable outsourced decryption, based on attribute-based key encapsulation mechanism, symmetric-key encryption and commitment scheme decryption, which reduces the computation cost and the bandwidth by half. The outsourced decryption verification model of [44] can also be applied to CP-ABE. For example, Premkamal et al. [29] proposed a CP-ABE scheme that supports verifiable outsourced computing. This scheme limits the number of access requests a user can make, and it resists chosen-plaintext attacks, collusion attacks, and agent attacks, hence realizing big data privacy protection in cloud environments. Ge et al. [17] proposed an ABE scheme that supports reliable outsourced decryption. In this scheme, smart contracts are used to ensure that the decryption cloud server is rewarded when and only when it returns a correctly transformed ciphertext.

The above schemes reduce the computational cost on devices with the assistance of cloud computing. However, the total system computational cost is unchanged and remains very high since them all use bilinear pairing as the primary operation.

2.3. Pairing-Free ABE Schemes for IoT

In recent years, pairing-free ABE schemes have emerged as a significant research area to adapt lightweight ABE for resource-constrained devices in the IoT. Yao et al. [30] proposed an ECC-based ABE scheme and proved the security in an attribute-based choice set model under the ECDDH problem. This scheme aims to address the security and privacy of data in IoT, but has limitations in terms of access control granularity, extensibility and versatility. Odelu et al. [45] proposed an RSA-based CP-ABE scheme for cloud-based IoT applications. However, its use of an AND gate as the access structure makes it less expressive, restricting its application in scenarios requiring more complex access control policies. The pairing-free CP-ABE scheme for IoT proposed by Ding et al. [31] uses LSSS to achieve fine-grained access control. However, there are two challenges with this scheme. The first one is security, as the public key is not used for encryption, leading to the possibility that users who do not satisfy the access policy may be able to decrypt the ciphertext without the private key. The second one is feasibility, as the single fully trusted AA in the scheme needs to take on both the attribute management and outsourcing computing, which may result in overloading the AA with tasks, making the AA a bottleneck of the whole system.

Sowjanya et al. [46] proposed an ECC-based KP-ABE scheme, but it is not applicable to the IoT with a huge number of devices. Subsequently, Sowjanya and Dasgupta [32] designed an ECC-based CP-ABE scheme to handle private data in the IoT-based healthcare system. However, this scheme lacks scalability and versatility which makes it difficult to migrate to other scenarios. There are also some ECC-based ABE schemes (e.g., [34,35,47]) have been proposed, but the computational cost of them is expensive. Wang et al. [33] proposed a pairing-free CP-ABE scheme supporting attribute revocation for cloud-assisted smart grids, which, unlike the scheme of [31], can resist illegal key sharing attacks. However, this scheme only supports access tree, which reduces the flexibility of access control. In addition, the decryption cost and communication cost are significantly increased. Sun et al.

All of the above schemes construct lightweight ABE schemes by removing bilinear pairings, but they all have some shortcomings and thus need further improvement to make them truly effective pairing-free ABE schemes.

3. Preliminaries

3.1. Linear Secret Sharing Scheme

In ABE schemes, the data owner needs to set up access structures in order to control the access rights of other users. Available access structures include monotonic Boolean formulas, AND gate, access tree and LSSS. Each access structure has its own specific characteristics and advantages. Choosing an appropriate access structure can facilitate the targeted optimization and enhancement of the ABE scheme, in terms of computational speed, storage density, expression flexibility and control granularity. LSSS is one of the most commonly chosen access structures for recent CP-ABE schemes. Therefore, a brief description of LSSS is given here.

Denote the set of all integers from 0 to $p-1$ as $Z_p = \{0, 1, \cdots, p-1\}$. Let $P = \{P_1, P_2, \cdots, P_n\}$ be a set of entities and Π be a secret sharing scheme over P. Π is an LSSS if it satisfies the following conditions.

(1) There exists a $l \times n$ shared matrix Λ for Π. Denote the i-th row vector of Λ as Λ_i. Each row of Λ corresponds to a mapping $\rho(i)$ from the set $\{1, 2, \cdots, l\}$ onto P. Suppose that the secret is $s \in Z_p$. Randomly select $r_2, r_3, \cdots, r_n \in Z_p$ and construct the column vector $v = (s, r_2, r_3, \cdots, r_n)^{\mathrm{T}}$. Then $\Lambda_i v$ is the l secret component generated by s according to Π, and $\Lambda_i v$ corresponds to the entity $\rho(i)$.

(2) All the components $\Lambda_i v$ form a one-dimensional vector on Z_p.

LSSS has linear reconfigurability. Suppose that Π is an LSSS for the access structure A. Given an authorized set $S \in \mathrm{A}$ and $I = \{i : \rho(i) \in S\} \subset \{1, 2, \cdots, n\}$, then there exists a set of constants $c = \{c_i \in Z_p\}_{i \in I}$ that can be found in polynomial time, such that $\sum_{i \in I} c_i \Lambda_i = (1, 0, \cdots, 0)$. Furthermore, the secret s is recovered by $\sum_{i \in I} c_i \Lambda_i v = (1, 0, \cdots, 0)(s, r_2, r_3, \cdots, r_n)^{\mathrm{T}} = s$. If a set $S \notin \mathrm{A}$ is given, then there does not exist a matching set of constants.

To make it easier for the reader to understand how matrix Λ enables flexible access control, an example is given here with reference to [30,31]. As shown in Figure 1, an access structure $\mathrm{A} = (\mathrm{A}_1 \text{ OR } \mathrm{A}_2)\mathrm{AND}(\mathrm{A}_3 \text{ OR } \mathrm{A}_4)$ can be transformed into a 4×2 LSSS matrix Λ.

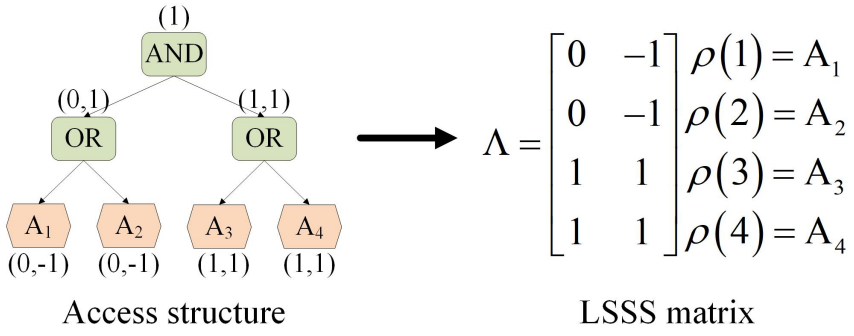

Figure 1. An example of the LSSS matrix representation access structure.

3.2. Formal Structure of CP-ABE

The standard CP-ABE consists of four algorithms, which are defined as follows.

(1) $Setup(\lambda, U) \rightarrow (PK, MSK)$: This algorithm is run by the AA, with the input security parameter λ and the full set of attributes U, to produce the output public parameter PK and the master secret key MSK.

(2) $KeyGen(S, MSK) \rightarrow SK$: This algorithm is also operated by the AA and is responsible for generating the decryption key for all the users in the system. The algorithm inputs the set of user attributes S and the master secret key MSK, and outputs the secret key SK.

(3) $Enc((\Lambda, \rho), PK, m) \rightarrow CT$: The data owner encrypts plaintext by this algorithm. The algorithm inputs the access structure (Λ, ρ), the public parameter PK and the plaintext m, and outputs the ciphertext CT.
(4) $Dec(CT, PK, SK) \rightarrow m$ or $\perp$: The user decrypts the ciphertext by this algorithm. The algorithm inputs the ciphertext CT, the public parameters PK and the secret key SK. If the set of attributes S satisfies the access structure (Λ, ρ), the decryption succeeds and the plaintext m is outputted. Otherwise the decryption fails and the terminator $\perp$ is outputted.

3.3. Elliptic Curve Decisional Diffie-Hellman Problem

Definition 1. *Let $\mathbb{G}$ be a cyclic group and G is a generator of $\mathbb{G}$. Given elements G, aG, bG and Z, where $a, b \in \mathbb{G}$ and Z may be equal to abG or a random value in $\mathbb{G}$. The output is a judgment for $Z = abG$.*

The advantage of an algorithm $\mathcal{B}$ to solve the ECDDH problem can defined as

$$\begin{aligned} \mathrm{Adv}(\mathcal{B}) =& \frac{1}{2}\Pr[\mathcal{D}(G, aG, bG, \mathrm{Z} = abG) = 1] \\ &+ \frac{1}{2}\Pr[\mathcal{D}(G, aG, bG, \mathrm{Z} = R) = 1] - \frac{1}{2}. \end{aligned} \tag{1}$$

4. System Architecture

4.1. System Model

The proposed system model is depicted in Figure 2, which involves six entities, namely, central authority (CA), attribute authorities (AAs), data owner (DO), data user (DU), cloud service provider (CSP), and edge server (ES). The main functions of each entity are described below.

(1) CA: A fully trusted certification authority. It is unconditionally trusted by all the users, and is responsible for establishing the system and setting the global parameters. All the AAs and users must apply for registration with the CA. After successful registration, the CA will assign a globally unique identity to each of them. The CA will not be involved in the management of user attributes and distribution of keys during the process of data sharing.
(2) AAs: A set of fully trusted attribute authorities with the responsibility of distributing, updating and revoking attributes. In the system, IoT devices are users, and AAs set user attributes for them based on their identities and tasks. Our scheme adopts multi-authority to jointly manage user attributes, and none of the attributes managed by an AA overlaps with other AAs. During the initialization phase of the whole system and when a new device is connected to the system, each AA generates an attribute public–private key pair for the device based on the set of user attributes that it manages.
(3) DO: The owner of the data. DO creates an access policy that fits its security requirements based on the attribute fields defined in the system. The data that needs to be shared is then encrypted according to the access policy. Finally, the ciphertext is uploaded to the cloud and stored by the cloud server.
(4) DU: A requester of data. DU is a legitimate user in the system with an identity assigned by the CA and a set of attributes distributed by the AAs. DU can submit an access request to the CSP to download the ciphertext, and decrypt the ciphertext with the assistance of ES. Only the DUs that satisfy the access policy configured by the DO can successfully decrypt the ciphertext. However, DUs are not fully trustworthy. When the ciphertext cannot be decrypted independently, a DU may communicate privately with other users, team up with several users to assemble a set of attributes, and eventually launch a collusion attack in order to obtain the data.

(5) CSP: A third-party CSP that is not fully trusted. CSP has powerful computing resources and abundant storage space, and it provides some services to customers according to cooperation agreements and laws and regulations. In our scheme, CSP is the cloud side of the IoT, responsible for providing data storage services and access control services for IoT devices. DO can upload data that it cannot store and data that needs to be shared to CSP. DU has the right to apply for access to the CSP and download data after confirming their legal identity. Although CSP should protect the privacy of users while providing services, it may use data in order to analyze user portraits and mine valuable information. In addition, CSP may also be attacked by criminals, resulting in data leakage and tampering. Therefore, it is necessary to encrypt the data before uploading it to the CSP to ensure the security of private information.

(6) ES: ES does not have as rich storage space as CSP, but has higher computing power and greater energy supply than IoT devices, and is responsible for assisting DU in decrypting the ciphertext. DU sends a decryption request to ES and provides the transformation key, then ES downloads the relevant ciphertext from CSP and decrypts it. As ES can only perform most of the complex computation work in the decryption process, it cannot fully decrypt the ciphertext. Therefore, ES is unable to obtain the plaintext. The whole process transforms the ciphertext without revealing any data, and reduces the computational cost of IoT device.

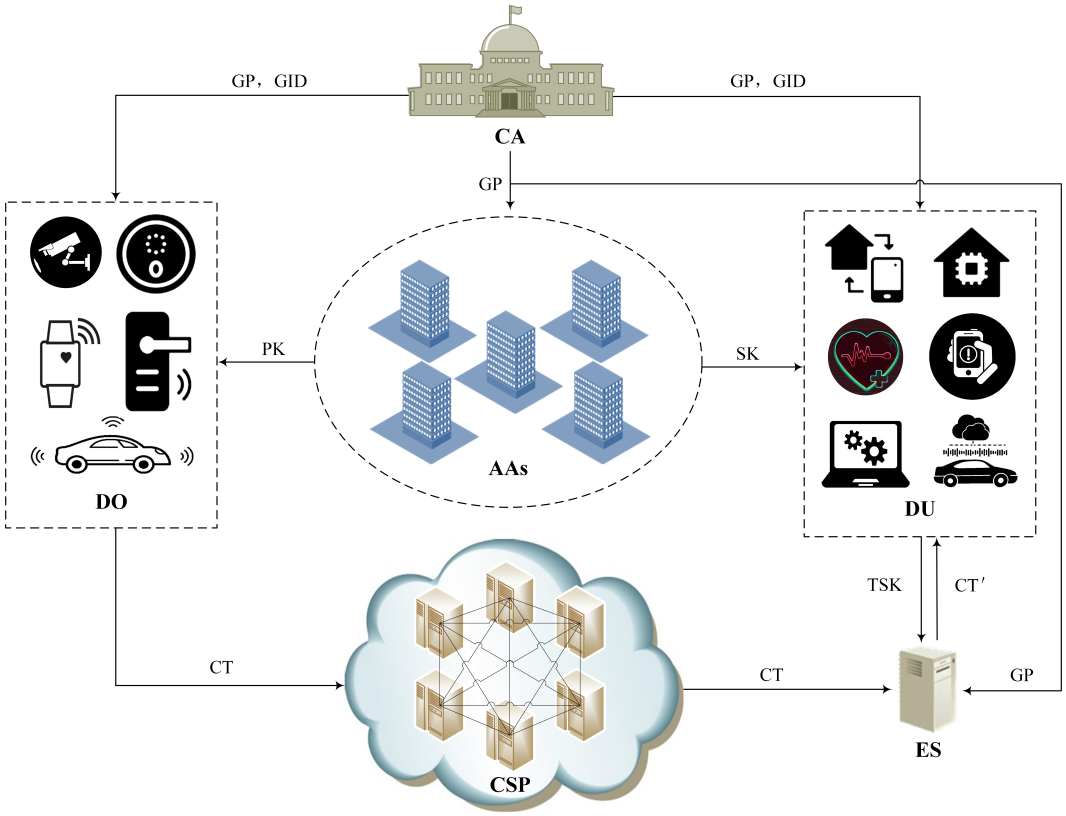

Figure 2. System model.

4.2. The Overview of Proposed Scheme

An overview of the proposed scheme is depicted in Figure 3, which includes the seven algorithms of *GlobalSetup*, *AuthoritySetup*, *KeyGen*, *KeyCon*, *Enc*, *EdgeDec*, and *Dec*. A description of each algorithm is given below.

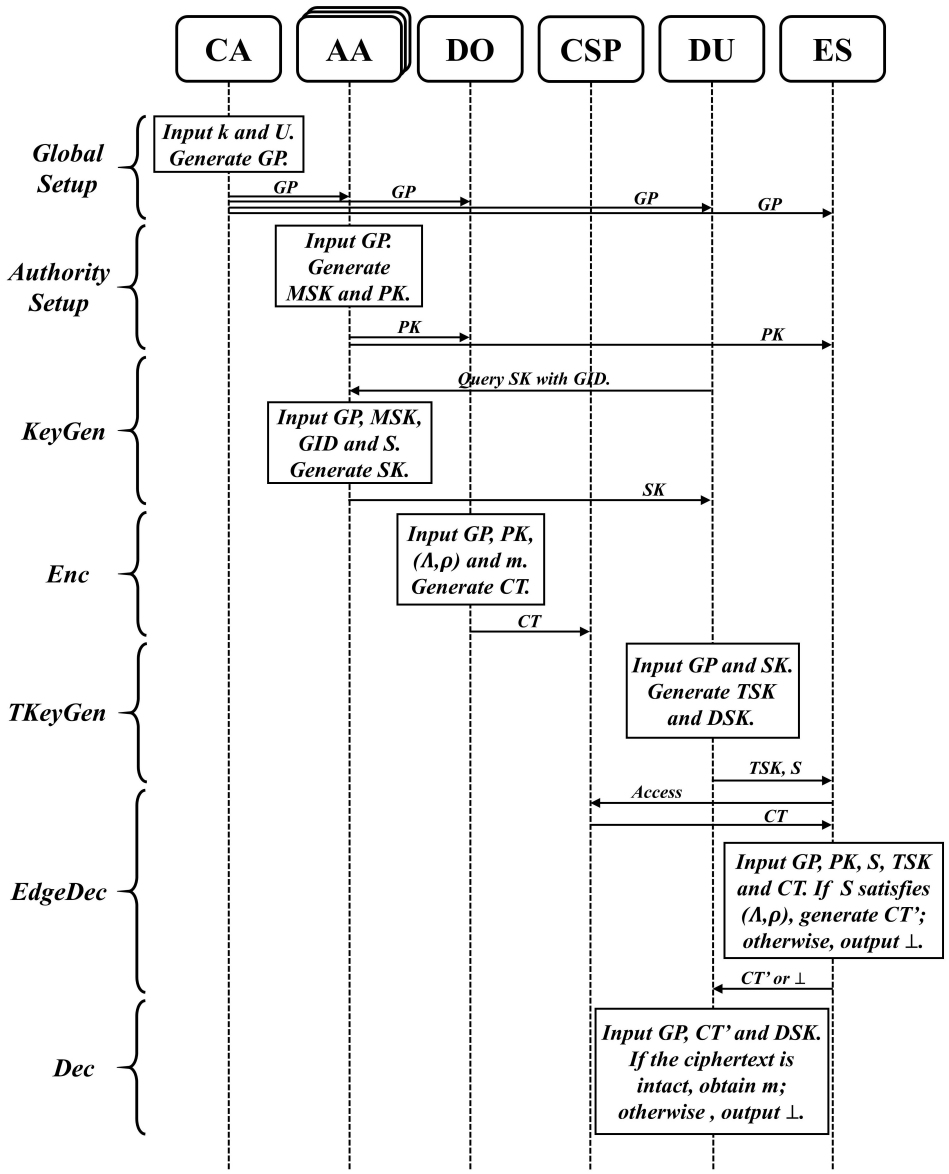

Figure 3. Overview of the proposed scheme.

(1) $GlobalSetup(k, U) \rightarrow GP$: The $GlobalSetup$ algorithm is operated by CA, which inputs the system security parameters k and the full set of system attributes U, and then outputs the global parameters GP.

(2) $AuthoritySetup(GP) \rightarrow (PK, MSK)$: Assume that there are n AAs in the system $\{AA_1, AA_2, \cdots, AA_n\}$, and AA_i manages the set of attributes U_i. In this phase, AA runs the $AuthoritySetup$ algorithm with the input system global parameters GP, to output the public key PK and the master private key MSK. The master private key MSK is stored by AA alone and the public key PK is published in the system.

(3) $KeyGen(GP, MSK, GID, S) \rightarrow SK$: The $KeyGen$ algorithm is operated by multiple AAs together. It inputs the global parameters GP, the master private key MSK, the user's identifier GID and the set of attributes S, and then outputs the secret key SK.

(4) $TKeyGen(GP, SK) \rightarrow (TSK, DSK)$: DU runs the $TKeyGen$ algorithm, inputting the global parameters GP and the secret key SK, and outputting the transformation key TSK and the decryption key DSK. ES can use TSK to perform an assisted decryption of the ciphertext, and then DU uses DSK to perform a final decryption.

(5) $Enc(GP, PK, (\Lambda, \rho), m) \rightarrow CT$: The Enc algorithm is run by DO. It inputs the global parameters GP, the public key PK, the access structure (Λ, ρ) and the plaintext m, and then outputs the ciphertext CT.
(6) $EdgeDec(GP, PK, S, TSK, CT) \rightarrow CT'$ or $\perp$: The $EdgeDec$ algorithm is operated by ES with the inputs of the global parameters GP, the public key PK, the set of attributes S, the transformation key TSK and the ciphertext CT. If S satisfies (Λ, ρ), it outputs the transformed ciphertext CT'. Otherwise, it outputs the termination symbol $\perp$.
(7) $Dec(GP, CT', DSK) \rightarrow m$ or $\perp$: The Dec algorithm is operated by DU with the inputs of the global parameters GP, the transformed ciphertext CT' and the decryption key DSK. If the integrity of the ciphertext is correct, the plaintext m is outputted. Otherwise the termination symbol $\perp$ is outputted.

4.3. Security Model

The indistinguishability under chosen plaintext attack (IND-CPA) security of a CP-ABE scheme is commonly proved by an attack game between the adversary and the challenger [30,31,33]. The six phases of this game are presented as follows.

(1) Initialization: The adversary selects an access structure (Λ, ρ) to hand to the challenger.
(2) Setup: The challenger runs the $GlobalSetup$ algorithm, generating the global parameters GP. Then the $AuthoritySetup$ algorithm is run to generate the public key PK and the master private key MSK. Finally, GP and PK are sent to the adversary.
(3) Phase 1: The adversary selects a global identity GID and then acquires a legitimate set of attributes S from trusted AAs. However, it is required that none of the attributes in S satisfies the access structure (Λ, ρ). Then the adversary submits $\{GID, S\}$ to the challenger and initiates a secret key query. After receiving $\{GID, S\}$, the challenger operates the $KeyGen$ algorithm and returns the generated secret key SK to the adversary.
(4) Challenge: The adversary creates 2 equal length plaintexts m_0 and m_1 and submits $\{m_0, m_1\}$ to the challenger. The challenger flips a random coin $\beta \in \{0, 1\}$ and runs the Enc algorithm to encrypt the plaintext m_β according to the access structure (Λ, ρ). Finally it send the produced ciphertext CT to the adversary.
(5) The adversary repeats the steps in Phase 1.
(6) Guess: Based on the ciphertext CT, the adversary determines which of the plaintexts m_0 and m_1 the challenger has selected for encryption and gives a guess $\beta' \in \{0, 1\}$. The adversary wins the game when $\beta' = \beta$.

In this game, $\Pr[\beta' = \beta]$ represents the probability of a correct guess, sand hence $\Pr[\beta' = \beta] - \frac{1}{2}$ represents the opponent's advantage in this game.

Definition 2. *The scheme is chosen-plaintext attack secure if any polynomial-time adversary has at most a negligible advantage to win the above attack game.*

5. Proposed Scheme

We now provide the full technical details of our proposed pairing-free CP-ABE scheme. The notations and descriptions used in this section are summarized in Table 1.

$GlobalSetup(k, U) \rightarrow GP$: After inputting the security parameter k and the full set of attributes U of the system, CA chooses a large prime p, and a finite field $GF(p)$ of order p according to k. Let E be an elliptic curve defined over the finite field $GF(p)$. Then, a point G of order r is chosen as a base point on E, and a cyclic group $\mathbb{G}$ on E is generated from G. Note that the elliptic curve discrete logarithm problem (ECDLP) on $\mathbb{G}$ is unsolvable in polynomial time. A strongly collision-resistant hash function $H : \{0, 1\}^* \rightarrow Z_r^*$ is chosen, where $Z_r^* = Z_r \backslash \{0\}$, while $\{0, 1\}^*$ indicates that the input message to H can be any length. Finally, CA generates the global parameter as Equation (2) and publishes it.

$$GP = \{GF(p), E, G, U, H\} \tag{2}$$

Table 1. Abbreviations and notations.

Notation	Descriptions
k	Security parameter
U	attribute set of system
$\mathbb{G}$	cyclic group
G	Generator of $\mathbb{G}$
H	Hash function
GP	Global parameter
GID	Identifier
MSK	Master private key
x_i, y_i	Master private key components for attribute i
PK	Public key
x_iG, y_iG	Public key components for attribute i
$SK_{GID,i}$	Secret key of user GID for attribute i
sk	Secret value
ck	Symmetric key
E_{ck}	symmetric encryption algorithm with ck
m	Plaintext containing f data files
m_i	ith data file
M_i	Symmetric encrypted ciphertext of m_i
E_i	Verification data of M_i
u, v	n-dimensional vectors
CT	ABE ciphertext
$C_0, C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}, C_{5,i}$	Components of the ciphertext CT
TSK_i	Transformation key for attribute i
DSK	Decryption key
CT'	Transformation ciphertext

$AuthoritySetup(GP) \rightarrow (PK, MSK)$: In the system, there are n AAs, each managing a particular set of attributes independently, and there is no overlap between any two sets of attributes. Each AA takes the input GP to run the $AuthoritySetup$ algorithm. AA first chooses a pair of random values $x_i, y_i \in Z_r$ for each attribute i according to the set of attributes that it manages. Then, AA generates the master private key as Equation (3) and the public key as Equation (4). Finally, AA keeps MSK and makes PK public in the system.

$$MSK = \{x_i, y_i\}_{\forall i} \tag{3}$$

$$PK = \{x_iG, y_iG\}_{\forall i} \tag{4}$$

Additionally, AA maintains a list of attributes associated with the identifier GID for each legitimate user in the system.

$KeyGen(GP, MSK, GID, S) \rightarrow SK$: This algorithm is run by all the AAs, and each AA generates the secret key for the user based on the set of attributes that it manages. The user submits GID to AA and requests a secret key, and AA takes the inputs GP, MSK and GID to run the $KeyGen$ algorithm. AA first searches the list of attributes for the user based on GID. Then, AA generates the secret key as Equation (5) for each of the user's attributes using the individually saved MSK. Finally, AA records the $SK_{GID,i}$ according to the user's

GID and attribute i. When the user applies for the secret key again, AA will directly return the recorded $SK_{GID,i}$.

$$SK_{GID,i} = \{x_i, H(GID)y_i\}_{i \in S} \tag{5}$$

$Enc(GP, PK, (\Lambda, \rho), m) \rightarrow CT$: Algorithm 1 shows the detailed flow of the Enc algorithm. The proposed scheme adopts a hybrid encryption approach, namely, using symmetric key encryption for the plaintext of the data to be shared, followed by asymmetric key encryption for the symmetric key. As a result, more than one data with the same access structure can be shared at a time. DU can request access to part or all of the data depending on the access requirements. DO defines the LSSS access structure (Λ, ρ), which specifies the attributes of DU who can access the data. Since DUs that can access data are no longer specified with identities, but are described using attributes, any DU that satisfies the LSSS access structure defined by the DO is a potential accessor. Therefore, the DO only needs to encrypt the data once before it can be shared to different DUs. It runs the Enc algorithm, inputting GP, PK, (Λ, ρ) and the plaintext $m = \{m_1, m_2, \cdots, m_f\}$, where m includes f data files. DO picks a secret value $sk \in \mathbb{G}$ randomly and generates a symmetric key $ck = H(sk)$. Then according to ck and the symmetric encryption algorithm E_{ck}, DO generates the ciphertext $\left\{M_1, M_2, \cdots, M_f\right\}$ via Equation (6).

Algorithm 1 Enc

Input: $GP, PK, (\Lambda, \rho), m$.
Output: CT.

$\{m_1, m_2, \cdots, m_f\} = m$
$sk \in \mathbb{G}$
$ck = H(sk)$
for $i \in [1, f]$ **do**
    $M_i = E_{ck}(m_i, ck)$
    $E_i = H(M_i \parallel ck)$
end for
$s \in Z_r$
$C_0 = sk + sG$
$u = (s, u_2, u_3, \cdots, u_n) \in Z_r^n$, $v = (0, v_2, v_3, \cdots, v_n) \in Z_r^n$
$\lambda_i = \Lambda_i u, \omega_i = \Lambda_i v$
for $i \in [1, l]$ **do**
    $\alpha_i, \beta_i, \gamma_i \in Z_r$
    $C_{1,i} = \lambda_i - \alpha_i \gamma_i, C_{2,i} = \omega_i - \beta_i \gamma_i, C_{3,i} = \gamma_i G, C_{4,i} = (x_i + \alpha_i) C_{3,i}, C_{5,i} = (y_i + \beta_i) C_{3,i}$
end for
$CT = \left\{(\Lambda, \rho), \{M_i, E_i\}_{i \in \{1, \cdots, f\}}, C_0, \{C_{1,j}, C_{2,j}, C_{3,j}, C_{4,j}, C_{5,j}\}_{j \in \{1, \cdots, l\}}\right\}$
return CT

$$M_i = E_{ck}(m_i, ck), i \in [1, f] \tag{6}$$

In order for DU to verify the integrity of the data ciphertext and prevent the data ciphertext from being tampered and replaced, the verification data needs to be generated. For $\left\{M_1, M_2, \cdots, M_f\right\}$, DO computes the verification data $\left\{E_1, E_2, \cdots, E_f\right\}$ via Equation (7).

$$E_i = H(M_i \parallel ck), i \in [1, f] \tag{7}$$

To blind the secret value sk, DO select a random value $s \in Z_r$ and computes a component of the ciphertext $C_0 = sk + sG$. Let Z_r^n represent the set of n-dimensional vectors with all the elements of each vector belonging to Z_r. DO randomly chooses two vectors as Equation (8), whose first elements are s and 0, respectively.

$$u = (s, u_2, u_3, \cdots, u_n) \in Z_r^n$$
$$v = (0, v_2, v_3, \cdots, v_n) \in Z_r^n \tag{8}$$

Next, DO computes $\lambda_i = \Lambda_i u$ and $\omega_i = \Lambda_i v$, where $\forall i \in \{1, 2, \cdots, l\}$ and Λ_i is the ith row of the access control matrix. Afterward, DO randomly selects $\alpha_i, \beta_i, \gamma_i \in Z_r$ and computes the rest component of the ciphertext via Equation (9). Eventually, DO generates a complete ciphertext as Equation (10) and send it to CSP for storage.

$$\begin{aligned} C_{1,i} &= \lambda_i - \alpha_i \gamma_i \\ C_{2,i} &= \omega_i - \beta_i \gamma_i \\ C_{3,i} &= \gamma_i G \\ C_{4,i} &= (x_i + \alpha_i) C_{3,i} \\ C_{5,i} &= (y_i + \beta_i) C_{3,i} \end{aligned} \tag{9}$$

$$CT = \left\{ (\Lambda, \rho), \{M_i, E_i\}_{i \in \{1, \cdots, f\}}, C_0, \{C_{1,j}, C_{2,j}, C_{3,j}, C_{4,j}, C_{5,j}\}_{j \in \{1, \cdots, l\}} \right\} \tag{10}$$

$TKeyGen(GP, SK) \rightarrow (TSK, DSK)$: DU runs the $TKeyGen$ algorithm after obtaining the SK from AA. AA chooses the random number $z \in Z_r$ as the decryption key and generates the transformation key $TSK = \{TSK_i\}_{i \in S}$ as Equtaion (11). DU saves the decryption key $DSK = \{z\}$ alone for the final decryption phase to obtain the plaintext, sends the transformation key TSK to ES and entrusts ES with the assisted decryption of the ciphertext.

$$TSK_i = x_i + H(GID) y_i z \tag{11}$$

$EdgeDec(GP, PK, S, TSK, CT) \rightarrow CT'$ or $\perp$: After receiving TSK and S from DU, ES downloads the specified ciphertext CT from CSP.

If S satisfies (Λ, ρ), then a set of constants $c = \{c_1, c_2, \cdots, c_l\} \in Z_r^l$ can be found in polynomial time, which satisfies $\sum_{i \in S} c_i \Lambda_i = (1, 0, 0, \cdots, 0)$. Therefore, ES can perform Equation (12) to partially decrypt ciphertext and generate the transformed ciphertext as Equation (13). Finally, the transformed ciphertext CT' is sent to DU.

$$\begin{aligned} C'_{1,i} &= C_{1,i} G + C_{4,i} \\ C'_{2,i} &= H(GID)(C_{2,i} G + C_{5,i}) \\ C'_{3,i} &= C_{3,i} TSK_i \\ CT_1 &= \textstyle\sum_{i \in S} c_i C'_{1,i} \\ CT_2 &= \textstyle\sum_{i \in S} c_i C'_{2,i} \\ CT_3 &= \textstyle\sum_{i \in S} c_i C'_{3,i} \end{aligned} \tag{12}$$

$$CT' = \left\{ \{M_i, E_i\}_{i \in \{1, \cdots, f\}}, C_0, CT_1, CT_2, CT_3 \right\} \tag{13}$$

If S does not satisfy (Λ, ρ), the algorithm will terminate and send $\perp$ to DU.

$Dec(GP, CT', DSK) \rightarrow m$: DU first computes the secret value via Equation (14) after receiving CT'. Then, DU generates the symmetric key $ck = H(sk)$. Next, the integrity of $\left\{ M_1, M_2, \cdots, M_f \right\}$ is verified against $\left\{ E_1, E_2, \cdots, E_f \right\}$ to determine whether the data has been tampered with and replaced. Specifically, $\forall i \in \{1, 2, \cdots, f\}$, DU computes $E'_i = H(M_i \parallel ck)$ and compares it with E_i. If $E_i = E'_i$, which proves that M_i is complete, it

decrypts M_i by ck and recovers the plaintext m_i. If $E_i \neq E_i^{'}$, M_i is proved to be incomplete and $\perp$ is returned. There may be a number of incomplete data, for which DU only needs to make another access request, without having to repeatedly access the data it has obtained.

$$sk = C_0 - (CT_1 + zCT_2 - CT_3) \tag{14}$$

6. Security Analysis

6.1. Correctness Proof

In the proposed scheme, we define data plaintext $\left\{m_1, m_2, \cdots, m_f\right\}$, data ciphertext $\left\{M_1, M_2, \cdots, M_f\right\}$, verification data $\left\{E_1, E_2, \cdots, E_f\right\}$, access structure (Λ, ρ), ciphertext CT, attribute set S of DU, transformation key TSK and decryption key DSK. The correctness of the proposed scheme can be proved as follows.

First we verify the *EdgeDec* algorithm. Based on the ciphertext CT, ES can compute:

$$\begin{aligned} C_{1,i}^{\prime} =& C_{1,i}G + C_{4,i} = (\lambda_i - \alpha_i\gamma_i)G + (x_i + \alpha_i)C_{3,i} \\ =& (\lambda_i - \alpha_i\gamma_i)G + (x_i + \alpha_i)\gamma_i G = (\lambda_i + x_i\gamma_i)G, \end{aligned} \tag{15}$$

$$\begin{aligned} C_{2,i}^{\prime} =& H(GID)(C_{2,i}G + C_{5,i}) \\ =& H(GID)((\omega_i - \beta_i\gamma_i)G + (y_i + \beta_i)C_{3,i}) \\ =& H(GID)((\omega_i - \beta_i\gamma_i)G + (y_i + \beta_i)\gamma_i G) \\ =& H(GID)(\omega_i + y_i\gamma_i)G, \end{aligned} \tag{16}$$

$$C_{3,i}^{\prime} = C_{3,i}TSK_i = (x_i + H(GID)y_i z)\gamma_i G. \tag{17}$$

If S satisfies (Λ, ρ), then $\exists c = (c_1, c_2, \cdots, c_l) \in Z_r^l$ such that $\sum_{i \in S} c_i\Lambda_i = (1, 0, 0, \cdots, 0)$ Then:

$$\begin{aligned} \sum_{i \in S} c_i\lambda_i =& \sum_{i \in S} c_i\Lambda_i u \\ =& (1, 0, 0, \cdots, 0) \cdot (s, u_2, u_3, \cdots, u_n)^{\mathrm{T}} = s, \end{aligned} \tag{18}$$

$$\begin{aligned} \sum_{i \in S} c_i\omega_i =& \sum_{i \in S} c_i\Lambda_i vs. \\ =& (1, 0, 0, \cdots, 0) \cdot (0, v_2, v_3, \cdots, v_n)^{\mathrm{T}} = 0. \end{aligned} \tag{19}$$

ES next computes:

$$\begin{aligned} CT_1 =& \sum_{i \in S} c_i C_{1,i}^{\prime} = \sum_{i \in S} c_i(\lambda_i + x_i\gamma_i)G \\ =& \sum_{i \in S} c_i\lambda_i G + \sum_{i \in S} c_i x_i\gamma_i G = sG + \sum_{i \in S} c_i x_i\gamma_i G, \end{aligned} \tag{20}$$

$$\begin{aligned} CT_2 =& \sum_{i \in S} c_i C_{2,i}^{\prime} = \sum_{i \in S} c_i H(GID)(\omega_i + y_i\gamma_i)G \\ =& H(GID)\left(\sum_{i \in S} c_i\omega_i G + \sum_{i \in S} c_i y_i\gamma_i G\right) \\ =& H(GID)\sum_{i \in S} c_i y_i\gamma_i G, \end{aligned} \tag{21}$$

$$\begin{aligned} CT_3 =& \sum_{i \in S} c_i C_{3,i}^{\prime} = \sum_{i \in S} c_i(x_i + H(GID)y_i z)\gamma_i G \\ =& \sum_{i \in S} c_i x_i\gamma_i G + H(GID)\sum_{i \in S} c_i y_i\gamma_i zG. \end{aligned} \tag{22}$$

Next, we verify the *Dec* algorithm. According to CT', DU can compute:

$$\begin{aligned} C_0 - (CT_1 + zCT_2 - CT_3) &= (sk + sG) \\ &- \left(\left(sG + \sum_{i \in S} c_i x_i \gamma_i G \right) + zH(GID) \sum_{i \in S} c_i y_i \gamma_i G \right. \\ &- \left. \left(\sum_{i \in S} c_i x_i \gamma_i G + H(GID) \sum_{i \in S} c_i y_i \gamma_i z G \right) \right) \\ &= sk + sG - sG = sk. \end{aligned} \tag{23}$$

ck can be mapped by the elliptic curve E and the point sk. DU computes $E_i^{'} = H(M_i \parallel ck)$, $\forall i \in \{1, 2, \cdots, f\}$. If the data ciphertext is integral, $E_i^{'} = E_i$. Finally, M_i is decrypted using ck to get m_i. By now, DU successfully accesses to the information.

6.2. Security Proof

We next prove that the proposed scheme achieves the CPA security under the ECDDH problem.

Theorem 1. *Since the ECDDH problem is hard to solve, no polynomial-time adversary can break our scheme.*

Proof. First, if there exists an adversary $\mathcal{A}$ that can break our scheme in polynomial time with a non-negligible advantage $\varepsilon > 0$, then there exists an effective algorithm $\mathcal{B}$ that can distinguish an ECDDH tuple from a random tuple in polynomial time with the advantage $\frac{\varepsilon}{2} > 0$. In the attack game, algorithm $\mathcal{B}$ will be constructed as a simulator $\mathcal{D}$.

Challenger $\mathcal{C}$ selects an elliptic curve E defined over a finite field $GF(p)$, where the order of $GF(p)$ is p. The point G on E is chosen as the base point and the order of G is r. A cyclic group $\mathbb{G}$ on E is generated from G, and the elliptic curve discrete logarithm problem (ECDLP) in $\mathbb{G}$ is impossible to solve in polynomial time. Then, challenger $\mathcal{C}$ randomly chooses $\beta \in \{0, 1\}$, $a, b \in Z_r$ and $R \in \mathbb{G}$. If $\beta = 0$, set (G, aG, bG, abG) to the tuple (G, aG, bG, Z); Otherwise, set (G, aG, bG, R) to the tuple (G, aG, bG, Z). Then the tuple (G, aG, bG, Z) is delivered to simulator $\mathcal{D}$, and simulator $\mathcal{D}$ interacts with adversary $\mathcal{A}$ in the following game.

Initialization: Adversary $\mathcal{A}$ sets up an access structure (Λ, ρ) and sends it to simulator $\mathcal{D}$. (Λ, ρ) is to be challenged.

Setup: Simulator $\mathcal{D}$ first runs the $GlobalSetup(k, U)$ algorithm in the original scheme with a security parameter k and a full set of attributes U provided by challenger $\mathcal{C}$ to generate the global parameter $GP = \{GF(p), E, G, U, H\}$. Then, simulator $\mathcal{D}$ selects $x_i, y_i \in Z_r$ for each attribute $i \in U$, and set the public key $PK = \{x_i aG, y_i aG\}_{i \in U}$. Finally, GP and PK are given to adversary $\mathcal{A}$, and the master private key $MSK = \{x_i, y_i\}_{i \in U}$ is kept by simulator $\mathcal{D}$ alone.

Phase 1: Adversary $\mathcal{A}$ randomly chooses a GID, and then applies valid attributes to any number of all trusted AAs to make up its own attribute set S. However, it requires that adversary $\mathcal{A}$ should avoid the attributes used in (Λ, ρ) when choosing attributes, in order to restrict that there are no attributes in S that can satisfy (Λ, ρ). Simulator $\mathcal{D}$ then starts accepting queries for the secret key initiated by adversary $\mathcal{A}$. Simulator $\mathcal{D}$ responds according to the attributes corresponding to adversary $\mathcal{A}$'s GID in the attribute list and returns the generated secret key SK to adversary $\mathcal{A}$.

Challenge: Adversary $\mathcal{A}$ generates two plaintexts of equal length, m_0 and m_1, and sends them to $\mathcal{D}$. Simulator $\mathcal{D}$ chooses two random vectors $u = (s, u_2, u_3, \cdots, u_n) \in Z_r^n$ and $v = (0, v_2, v_3, \cdots, v_n) \in Z_r^n$ whose first elements are the cryptographic index s and 0, respectively. $\forall i \in \{1, 2, \cdots, l\}$, it computes $\lambda_i = \Lambda_i u$ and $\omega_i = \Lambda_i v$, where Λ_i is the ith row of the access control matrix. Then $\mathcal{D}$ randomly selects $\beta \in \{0, 1\}$ and encrypts the plaintext m_β according to (Λ, ρ) to generate the ciphertext CT. Specifically, $\mathcal{D}$ computes C_0 with

the $Enc(GP, PK, (\Lambda, \rho), m_\beta)$ in the original scheme. Subsequently, $\mathcal{D}$ selects $\alpha_i, \beta_i, \gamma_i \in Z_r$ and sets $C_{1,i} = \lambda_i - \alpha_i\gamma_i$, $C_{2,i} = \omega_i - \beta_i\gamma_i$, $C_{3,i} = \gamma_i Z$, $C_{4,i} = \gamma_i x_i aG + \alpha_i\gamma_i bG$ and $C_{5,i} = \gamma_i y_i aG + \beta_i\gamma_i bG$. Finally, CT is sent to adversary $\mathcal{A}$.

Phase 2: Repeat the operation of Phase 1 under the same constraints.

Guess: Adversary $\mathcal{A}$ chooses $\beta' \in \{0, 1\}$ as a guess for β.

If $\beta' = \beta$, simulator $\mathcal{D}$ outputs 1 to represent the guess result of $Z = abG$. In this case, the the adversary $\mathcal{A}$ successfully guesses the plaintext m_β, and thus wins the attack game. Otherwise, $\mathcal{D}$ outputs 0 to represent the guess result of $Z = R$.

Here, the probability of adversary $\mathcal{A}$ winning is defined as $\Pr[\beta' = \beta]$, and its advantage is given by $\Pr[\beta' = \beta] - \frac{1}{2}$.

When $Z = abG$, since adversary $\mathcal{A}$ has the advantage of $\varepsilon > 0$, the probability that simulator $\mathcal{D}$ guesses correctly for β is

$$\Pr[\mathcal{D}(G, aG, bG, Z = abG) = 1] = \frac{1}{2} + \varepsilon. \tag{24}$$

When $Z = R$, since R is chosen randomly, adversary $\mathcal{A}$'s guess for β does not have any advantage. Simulator $\mathcal{D}$'s guess for β fits the Bernoulli distribution at this time, and the probability of a correct guess is

$$\Pr[\mathcal{D}(G, aG, bG, Z = R) = 1] = \frac{1}{2}. \tag{25}$$

Therefore, the probability that simulator $\mathcal{D}$ succeeds is

$$\begin{aligned} \mathrm{Adv}(\mathcal{D}) =& \frac{1}{2}\Pr[\mathcal{D}(G, aG, bG, Z = abG) = 1] \\ &+ \frac{1}{2}\Pr[\mathcal{D}(G, aG, bG, Z = R) = 1] - \frac{1}{2} \\ =& \frac{1}{2}\left(\frac{1}{2} + \varepsilon\right) + \frac{1}{2} \times \frac{1}{2} - \frac{1}{2} = \frac{\varepsilon}{2}. \end{aligned} \tag{26}$$

Thus, simulator $\mathcal{D}$ distinguishes an ECDDH tuple from a random tuple by the advantage of $\varepsilon/2$. Because ε is non-negligible, the advantage $\frac{\varepsilon}{2}$ of $\mathcal{D}$ is also non-negligible.

However, there exists no effective algorithm that can solve the ECDDH problem, and thus adversary $\mathcal{A}$ does not exist. Therefore, the proposed scheme has the indistinguishability under chosen-plaintext attack (IDN-CPA). □

6.3. Resistant to Collusion Attack

In order to guarantee effective access control to the data, the scheme must be able to resist collusion attacks. In other words, more than one user cannot decrypt the ciphertext independently because their attributes does not satisfy the access structure (Λ, ρ). When these users collude with each other to piece together the attribute sets, they still cannot successfully decrypt the ciphertext. In the proposed scheme, CA binds a GID for each legitimate user in the system, and AA associates the attribute list with GID, so that different users cannot piece together attributes with each other for decryption because of different GID in the decryption process.

For example, consider two legitimate users, Alice and Bob, in the system, and a ciphertext associated with access structure $attr_\varphi \wedge attr_\psi$. Alice has the attribute $attr_\varphi$ and Bob has the attribute $attr_\psi$. Obviously, both Alice and Bob do not satisfy the access structure and cannot decrypt the ciphertext independently. Therefore, Alice and Bob collude together

and try to access the ciphertext by sharing attributes. During the decryption process, $\forall i \in \{attr_\varphi, attr_\psi\}$, Alice will get the final ciphertext computed by ES as follows:

$$\begin{cases} C'_{1,i,Alice} = \lambda_i G + x_i \gamma_i G, \\ C'_{2,i,Alice} = H(GID_{Alice})\omega_i G + H(GID_{Alice}) y_i \gamma_i G, \\ C'_{3,i,Alice} = x_i \gamma_i G + H(GID_{Alice}) y_i \gamma_i z G. \end{cases} \tag{27}$$

Bob will get the final ciphertext computed by ES as follows:

$$\begin{cases} C'_{1,i,Bob} = \lambda_i G + x_i \gamma_i G, \\ C'_{2,i,Bob} = H(GID_{Bob})\omega_i G + H(GID_{Bob}) y_i \gamma_i G, \\ C'_{3,i,Bob} = x_i \gamma_i G + H(GID_{Bob}) y_i \gamma_i z G. \end{cases} \tag{28}$$

If a user has attributes that satisfy the access structure $attr_\varphi \wedge attr_\psi$, a set of constants c can be computed in polynomial time to enable linear reconstruction. However, because Alice and Bob have different identifiers GID, it results in $H(GID_{Alice}) \neq H(GID_{Bob})$. This difference prevents them from colluding to compute sG, thereby ensuring they cannot decrypt the ciphertext together. Therefore, our scheme can effectively resists collusion attacks.

6.4. Data Confidentiality

AA chooses a pair of random values $x_i, y_i \in Z_r$ for each attribute i as the private key for attribute i. Then it generates the master private key $MSK = \{x_i, y_i\}_{\forall i}$ and the public key $PK = \{x_i G, y_i G\}_{\forall i}$. MSK is kept by AA alone. Since there does not exist an efficient algorithm that can break ECDLP in polynomial time, any user who has PK cannot obtain additional information about MSK. After encrypting the data plaintext m into data ciphertext M by symmetric key ck, DO maps ck to a point sk on the elliptic curve E and hides sk in $C_0 = sk + sG$. Since the blinding index $s \in Z_r$ is chosen randomly and kept by DO alone, it is hard for other users to conjecture the blinding factor sG. So other users are unable to separate sG from C_0, and cannot get sk, let alone decrypt M. DO construct s in the random vector $u = (s, u_2, u_3, \cdots, u_n) \in Z_r^n$. Then, according to the set access structure (Λ, ρ), u is divided into $\lambda_i = \Lambda_i u$, $i \in \{1, 2, \cdots, l\}$ by LSSS. In the decryption process, DU that can find an appropriate set of constants $c = \{c_1, c_2, \cdots, c_l\} \in Z_r^l$ in polynomial time to satisfy $\sum_{i \in S} c_i \Lambda_i = (1, 0, 0, \cdots, 0)$, making $\sum_{i \in S} c_i \lambda_i = s$, must have a set of attributes that meet (Λ, ρ). However, other DUs, who do not have the attributes meeting (Λ, ρ), cannot recover s in polynomial time, and consequently cannot continue to decrypt.

7. Performance Analysis

7.1. Comparison of Features

In order to analyze the achievable performance of our proposed scheme, we first compare its features with those of the existing schemes [19,30,31,33,41] in Table 2. Features include three categories: infrastructure, lightweightness and security. The infrastructure includes access structures and multiple authorities. Lightweightness compares whether the scheme uses a pairing-free design, supports outsourced decryption and the verifiability of outsourced decryption. Security compares resistance to collusion attacks and provable security.

As shown in Table 2, the schemes [19,31,41] support LSSS access structure and have more fine-grained access control than the schemes with access tree [30,33]. The schemes [19,41] implement multi-authority, but both use expensive bilinear pairing operations. Additionally, the scheme [19] is under the static assumption, while the scheme [41] is under the assumption of the q-decisional parallel bilinear Diffie–Hellman exponent 2 problem (q-DPBDHE2). The schemes [30,31,33] are pairing-free lightweight schemes that use elliptic curve scalar multiplication as the base operation. Moreover, they are IND-CPA under the ECDDH assumption. The scheme [31] supports LSSS access structure and

outsourced decryption, which can effectively reduce the computational cost of DU while providing fine-grained access control. However, this scheme does not support verifiable outsourced decryption, lacks multi-authority support, and faces the risk of illegal key sharing. The scheme [33] supports an access tree structure and resists collusion attacks, but it does not include multi-authority support or outsourced decryption capabilities. Among all the schemes compared in Table 2, only our scheme possesses all the desired features, namely, avoiding bilinear pairing, supporting multi-authority, verifiable outsourced decryption and LSSS access structure, resisting collusion attacks, and being IDN-CPA under the ECDDH assumption.

Table 2. Comparison of features for various schemes.

Scheme	Infrastructure		Lightweightness			Security	
	Access Structure	Multi-Authority	Pairing Free	Outsourced Decryption	Verifiable Outsourcing	Resistant to Collusion Attack	Provable Security
Lewko [19]	LSSS	✓	✗	✗	–	✓	fully security
Zhang [41]	LSSS	✓	✗	✗	–	✗	IND-CPA
Yao [30]	Access tree	✗	✓	✗	–	✗	IND-CPA
Ding [31]	LSSS	✗	✓	✓	✗	✓	IND-CPA
Wang [33]	Access tree	✗	✓	✗	–	✓	IND-CPA
Our Scheme	LSSS	✓	✓	✓	✓	✓	IND-CPA

7.2. Comparison of Computation Costs

We next compare the computation costs of our scheme with those of the existing schemes [19,30,31,33,41]. Computational complexity of a scheme is mainly concerned with the major operations performed during encryption and decryption. The basic operations include bilinear pairing operation, denoted as P, which is the most expensive operation, two types of modulo power operation, denoted as E and E_T, and scalar multiplication operation, denoted as S. Compared with these three operations, other lightweight operations can be ignored, such as arithmetic operations, point-addition operations, hashing operations, encryption and decryption of symmetric key encryption.

Table 3 compares the computation costs for the six schemes, where n is the number of all attributes in the system, l is the number of attributes in the access structure, and u is the number of attributes of the user. According to [30,32], scalar multiplication (S) can be used as the unit for comparing the computation costs. Since the bilinear pairing operations in the schemes [19,41] are symmetric, one bilinear pairing operation is approximately equal to three scalar multiplication operations. A modulo power operation is approximately equal to a scalar multiplication operation. According to the setting of the number of attributes in these comparing schemes, we assume $n = 30$, $l = 10$ and $u = 5$. Note that the encryption and decryption costs in the table are local costs and do not include computing costs outsourced to edge servers and the cloud.

Table 3. Comparison of computation costs for various schemes.

Scheme	Encryption	Decryption
Lewko [19]	$(5l+1)\,\mathsf{E} \approx 51\,\mathsf{S}$	$n\,\mathsf{E} + 2n\,\mathsf{P} \approx 210\,\mathsf{S}$
Zhang [41]	$6l\,\mathsf{E} + (2l+1)\,\mathsf{E}_\mathsf{T}$ $+(2l+1)\,\mathsf{P} \approx 144\,\mathsf{S}$	$3l\,\mathsf{E} + 3l\,\mathsf{P} \approx 120\,\mathsf{S}$
Yao [30]	$(l+1)\,\mathsf{S} \approx 11\,\mathsf{S}$	$(u+1)\,\mathsf{S} \approx 6\,\mathsf{S}$
Ding [31]	$(4l+1)\,\mathsf{S} \approx 41\,\mathsf{S}$	$(u+1)\,\mathsf{S} \approx 6\,\mathsf{S}$
Wang [33]	$(3l+2)\,\mathsf{S} \approx 32\,\mathsf{S}$	$(3l+1)\,\mathsf{S} \approx 31\,\mathsf{S}$
Our Scheme	$(3l+1)\,\mathsf{S} \approx 31\,\mathsf{S}$	$1\,\mathsf{S}$

As shown in Table 3, the computation costs of the schemes based on bilinear pairing [19,41] are much higher than the pairing-free schemes. Among the pairing-free schemes, the encryption cost of our scheme is slightly reduced compared with the schemes [31,33],

but is higher than that of the scheme [30]. For decryption, the number of scalar multiplication operations required in the schemes [30,31,33] is linearly related to the number of attributes, while our scheme only needs single scalar multiplication operation, which is the minimum and independent of the number of attributes.

In the IoT environment, as more devices are added, the attributes describing them must be refined to accommodate their diversity and heterogeneity. This will increase the number of attributes in the system, which, in turn, increases the decryption complexity of the schemes [19,30,31,33,41]. By contrast, the decryption complexity of our scheme remains to be the minimum of 1 S. Moreover, in the process of sharing one copy of data, the encryption algorithm needs to be performed only once by the data owner, while all the legitimate devices in the IoT are potential users, and any device that needs to access the data will need to decrypt the ciphertext. Consequently, the decryption algorithm will be performed many times. Therefore, the optimization of the decryption algorithm is more beneficial to reduce the overall computational burden of the system. Compared to other schemes [19,30,31,33,41], our scheme is much more efficient and better suited for the IoT environment.

8. Conclusions

In this paper, we proposed a secure and lightweight CP-ABE scheme without pairing, designed to facilitate controllable one-to-many data sharing among IoT devices. The proposed scheme sets multiple attribute authorities to decentralize attribute management, reducing the burden on a single attribute authority and avoiding the bottlenecks associated with single-point systems. To minimize computational complexity, the proposed scheme adopts a pairing-free construction, thereby avoiding the intensive computations required by bilinear pairing operations. Furthermore, by offloading most of the complex decryption tasks to an edge server, the decryption cost for data users is minimized, i.e., only one scalar multiplication operation needs to be performed. Thus, the proposed scheme is particularly suitable for IoT environments with limited computational resources. Security analysis has proved that the proposed scheme is IDN-CPA under the ECDDH assumption and resistant to collusion attacks. The performance analysis concluded that our scheme not only provides more features, but is also more computationally efficient than existing alternatives. In particular, while current unpaired attribute-based encryption schemes have linear decryption costs, our scheme achieves a constant decryption cost. In future work, we plan to extend this scheme to ensure the non-repudiation of cloud, edge, and user interactions through the integration of blockchain technology.

Author Contributions: Conceptualization , C.G. and M.W.; Methodology, C.G.; Software, C.G.; Validation, C.G.; Formal analysis, C.G. and S.C.; Investigation, S.C.; Resources, B.G.; Data curation, B.G. and S.T.; Writing – original draft, C.G., S.T. and S.C.; Writing – review & editing, B.G., M.W., H.A. and S.C.; Visualization, B.G., M.W., H.A. and S.C.; Supervision, B.G., M.W., H.A. and S.T.; Project administration, B.G., M.W., H.A. and S.T.; Funding acquisition, H.A. and S.T. All authors have read and agreed to the published version of this manuscript.

Funding: This work was supported in part by the Major Science and Technology Projects in Yunnan Province (202202AD080013), in part by the National Key Research and Development Project of China (2019YFB2102303), and National Natural Science Foundation of China (61971014). The authors also extend their appreciation to King Khalid University for funding this work through Large Group Project under grant number RGP.2/373/45.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sodhro, A.H.; Awad, A.I.; van de Beek, J.; Nikolakopoulos, G. Intelligent authentication of 5G healthcare devices: A survey. *Internet Things* **2022**, *20*, 100610. https://doi.org/https://doi.org/10.1016/j.iot.2022.100610.
2. Ahmed, S.F.; Alam, M.S.B.; Afrin, S.; Rafa, S.J.; Taher, S.B.; Kabir, M.; Muyeen, S.M.; Gandomi, A.H. Toward a Secure 5G-Enabled Internet of Things: A Survey on Requirements, Privacy, Security, Challenges, and Opportunities. *IEEE Access* **2024**, *12*, 13125–13145. https://doi.org/10.1109/ACCESS.2024.3352508.
   Dawood, M.; Tu, S.; Xiao, C.; Alasmary, H.; Waqas, M.; Rehman, S.U. Cyberattacks and Security of Cloud Computing: A Complete Guideline. Symmetry 2023, 15, 1981.
3. Dawood, M.; Tu, S.; Xiao, C.; Alasmary, H.; Waqas, M.; Rehman, S.U. Cyberattacks and Security of Cloud Computing: A Complete Guideline. *Symmetry* **2023**, *15*.
4. Khor, J.H.; Sidorov, M.; Ong, M.T.; Chua, S.Y. Public blockchain-based data integrity verification for low-power IoT devices. *IEEE Internet Things J.* **2023**, *10*, 13056–13064.
5. Revanesh, M.; Acken, J.M.; Sridhar, V. DAG block: Trust aware load balanced routing and lightweight authentication encryption in WSN. *Future Gener. Comput. Syst.* **2023**, *140*, 402–421.
6. Singh, S.; Sharma, P.K.; Moon, S.Y.; Park, J.H. Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions. *J. Ambient. Intell. Humaniz. Comput.* **2024**, *15*, 1625–1642.
7. Li, Y.; Li, Z.; Yang, B.; Ding, Y. Algebraic signature-based public data integrity batch verification for cloud-IoT. *IEEE Trans. Cloud Comput.* **2023**, *11*, 3184–3196.
8. Peng, S.; Zhao, L.; Al-Dubai, A.Y.; Zomaya, A.Y.; Hu, J.; Min, G.; Wang, Q. Secure Lightweight Stream Data Outsourcing for Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 10815–10829.
9. Jeyaraj, R.; Balasubramaniam, A.; MA, A.K.; Guizani, N.; Paul, A. Resource management in cloud and cloud-influenced technologies for internet of things applications. *ACM Comput. Surv.* **2023**, *55*, 1–37.
10. Hazra, A.; Donta, P.K.; Amgoth, T.; Dustdar, S. Cooperative transmission scheduling and computation offloading with collaboration of fog and cloud for industrial IoT applications. *IEEE Internet Things J.* **2022**, *10*, 3944–3953.
11. Purohit, K.C.; Manchanda, M.; Singh, A. Cloud Data Storage Security: The Challenges and a Countermeasure. In *Soft Computing: Theories and Applications: Proceedings of SoCTA 2020, Volume 1*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 97–105.
12. Li, L.; Cai, R. Research on Cloud Data Storage Security Privacy Protection System under Digital Campus. In Proceedings of the 2023 IEEE International Conference on Image Processing and Computer Applications (ICIPCA), Changchun, China, 11–13 August 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 314–319.
13. Backendal, M.; Davis, H.; Günther, F.; Haller, M.; Paterson, K.G. A formal treatment of end-to-end encrypted cloud storage. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2024; Springer: Berlin/Heidelberg, Germany, pp. 40–74.
14. Chauhan, M.; Shiaeles, S. An analysis of cloud security frameworks, problems and proposed solutions. *Network* **2023**, *3*, 422–450.
15. Chen, B.; Xiang, T.; He, D.; Li, H.; Choo, K.K.R. BPVSE: Publicly verifiable searchable encryption for cloud-assisted electronic health records. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 3171–3184.
16. Shruti.; Rani, S.; Sah, D.K.; Gianini, G. Attribute-based encryption schemes for next generation wireless IoT networks: A comprehensive survey. *Sensors* **2023**, *23*, 5921.
17. Ge, C.; Liu, Z.; Susilo, W.; Fang, L.; Wang, H. Attribute-based encryption with reliable outsourced decryption in cloud computing using smart contract. *IEEE Trans. Dependable Secur. Comput.* **2023**, *21*, 937–948.
18. Hou, Z.; Ning, J.; Huang, X.; Xu, S.; Zhang, L.Y. Blockchain-based efficient verifiable outsourced attribute-based encryption in cloud. *Comput. Stand. Interfaces* **2024**, *90*, 103854.
19. Lewko, A.; Waters, B. Decentralizing attribute-based encryption. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, 15–19 May 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 568–588.
20. He, Z.; Chen, Y.; Luo, Y.; Zhang, L.; Tang, Y. Revocable and Traceable Undeniable Attribute-Based Encryption in Cloud-Enabled E-Health Systems. *Entropy* **2023**, *26*, 45.
21. Wang, Y.; Pan, J.; Chen, Y. Fine-grained secure attribute-based encryption. *J. Cryptol.* **2023**, *36*, 33.
22. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 321–334.
23. Amit, S.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; Springer: Berlin, Germany, 2005; pp. 457–473.
24. Peñuelas-Angulo, A.; Feregrino-Uribe, C.; Morales-Sandoval, M. A revocable multi-authority attribute-based encryption scheme for fog-enabled IoT. *J. Syst. Archit.* **2024**, *155*, 103265.
25. Jiang, Y.; Xu, X.; Xiao, F. Attribute-based encryption with blockchain protection scheme for electronic health records. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 3884–3895.
26. Zhang, Z.; Zhang, W.; Qin, Z. A partially hidden policy CP-ABE scheme against attribute values guessing attacks with online privacy-protective decryption testing in IoT assisted cloud computing. *Future Gener. Comput. Syst.* **2021**, *123*, 181–195.
27. Feng, C.; Yu, K.; Aloqaily, M.; Alazab, M.; Lv, Z.; Mumtaz, S. Attribute-based encryption with parallel outsourced decryption for edge intelligent IoV. *IEEE Trans. Veh. Technol.* **2020**, *69*, 13784–13795.

28. Doshi, N.; Jinwala, D.C. Fully secure ciphertext policy attribute-based encryption with constant length ciphertext and faster decryption. *Secur. Commun. Netw.* **2014**, *7*, 1988–2002.
29. Premkamal, P.K.; Pasupuleti, S.K.; Alphonse, P. A new verifiable outsourced ciphertext-policy attribute based encryption for big data privacy and access control in cloud. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 2693–2707.
30. Yao, X.; Chen, Z.; Tian, Y. A lightweight attribute-based encryption scheme for the Internet of Things. *Future Gener. Comput. Syst.* **2015**, *49*, 104–112.
31. Ding, S.; Li, C.; Li, H. A novel efficient pairing-free CP-ABE based on elliptic curve cryptography for IoT. *IEEE Access* **2018**, *6*, 27336–27345.
32. Sowjanya, K.; Dasgupta, M. A ciphertext-policy Attribute based encryption scheme for wireless body area networks based on ECC. *J. Inf. Secur. Appl.* **2020**, *54*, 102559.
33. Wang, Y.; Chen, B.; Li, L.; Ma, Q.; Li, H.; He, D. Efficient and secure ciphertext-policy attribute-based encryption without pairing for cloud-assisted smart grid. *IEEE Access* **2020**, *8*, 40704–40713.
34. Sun, J.; Zhu, J.; Tian, Z.; Shi, G.; Guan, C. Attribute based encryption scheme based on elliptic curve cryptography and supporting revocation. *J. Comput. Appl.* **2022**, *42*, 2094.
35. Chandel, A.; Debnath, S.; Kumar, J.; Mohapatra, R.K. An ECC-Based Lightweight CPABE Scheme with Attribute Revocation. In Proceedings of the International Conference on Machine Learning, IoT and Big Data, Sarang, India, 10–12 March 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 505–515.
36. Boneh, D.; Franklin, M. Identity-based encryption from the Weil pairing. In Proceedings of the Annual international cryptology conference. Springer, Santa Barbara, CA, USA, 19–23 August 2001; pp. 213–229.
37. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 30 October–3 November 2006; pp. 89–98.
38. Ibraimi, L.; Tang, Q.; Hartel, P.; Jonker, W. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In Proceedings of the International conference on information security practice and Experience, Xi'an, China, 13–15 April 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 1–12.
39. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Proceedings of the International Workshop on Public Key Cryptography, Atlanta, GA, USA, 7–10 May 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 53–70.
40. Nishide, T.; Yoneyama, K.; Ohta, K. Attribute-based encryption with partially hidden encryptor-specified access structures. In Proceedings of the International Conference on Applied Cryptography and Network Security, New York, NY, USA, 3–6 June 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 111–129.
41. Zhang, K.; Li, H.; Ma, J.; Liu, X. Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability. *Sci. China Inf. Sci.* **2018**, *61*, 1–13.
42. Green, M.; Hohenberger, S.; Waters, B. Outsourcing the decryption of abe ciphertexts. In Proceedings of the USENIX Security Symposium, Francisco, CA, USA, 8–12 August 2011; Volume 2011.
43. Lai, J.; Deng, R.H.; Guan, C.; Weng, J. Attribute-based encryption with verifiable outsourced decryption. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 1343–1354.
44. Lin, S.; Zhang, R.; Ma, H.; Wang, M. Revisiting attribute-based encryption with verifiable outsourced decryption. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2119–2130.
45. Odelu, V.; Das, A.K.; Khan, M.K.; Choo, K.K.R.; Jo, M. Expressive CP-ABE scheme for mobile devices in IoT satisfying constant-size keys and ciphertexts. *IEEE Access* **2017**, *5*, 3273–3283.
46. Sowjanya, K.; Dasgupta, M.; Ray, S.; Obaidat, M.S. An efficient elliptic curve cryptography-based without pairing KPABE for Internet of Things. *IEEE Syst. J.* **2019**, *14*, 2154–2163.
47. Khasawneh, S.; Kadoch, M. ECS-CP-ABE: A lightweight elliptic curve signcryption scheme based on ciphertext-policy attribute-based encryption to secure downlink multicast communication in edge envisioned advanced metering infrastructure networks. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4102.