

Article

# Data Rights Confirmation Scheme Based on Auditable Ciphertext CP-ABE in the Cloud Storage Environment

Lingyun Zhang , Yuling Chen \* , Yun Luo, Zhongxiang He and Tao Li 

State Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang 550001, China; gs.lyzhang20@gzu.edu.cn (L.Z.); gs.yunluo20@gzu.edu.cn (Y.L.); zhongxiangdage@gmail.com (Z.H.)

\* Correspondence: ylchen3@gzu.edu.cn

**Abstract:** Advances in information technology have made data accessible anytime and anywhere. Currently, data confirmation is a popular area of research. Many current approaches to data confirmation rely on submitting certificates of ownership, embedding digital watermarks, or using blockchain. However, none of these approaches can avoid exposing source data to third parties that are not fully trusted. To address this issue, this paper proposes a new data confirmation method based on ciphertext policy attribute-based encryption (CP-ABE), which is widely used in cloud storage environments. The unique identifier of the data owner is encrypted by Paillier encryption and embedded into the ciphertext, so that the ownership corresponding to the plaintext is converted to the ownership corresponding to the ciphertext. During the entire confirmation process, third-party organizations cannot access the source data, reducing the risk of source data leakage. Finally, the feasibility of the scheme is proved by security proof and experiment comparison.

**Keywords:** CP-ABE; cloud storage; data confirmation; cryptography; Paillier encryption



**Citation:** Zhang, L.; Chen, Y.; Luo, Y.; He, Z.; Li, T. Data Rights Confirmation Scheme Based on Auditable Ciphertext CP-ABE in the Cloud Storage Environment. *Appl. Sci.* **2023**, *13*, 4355. <https://doi.org/10.3390/app13074355>

Academic Editor: Dimitris Mourtzis

Received: 9 March 2023

Revised: 20 March 2023

Accepted: 27 March 2023

Published: 29 March 2023

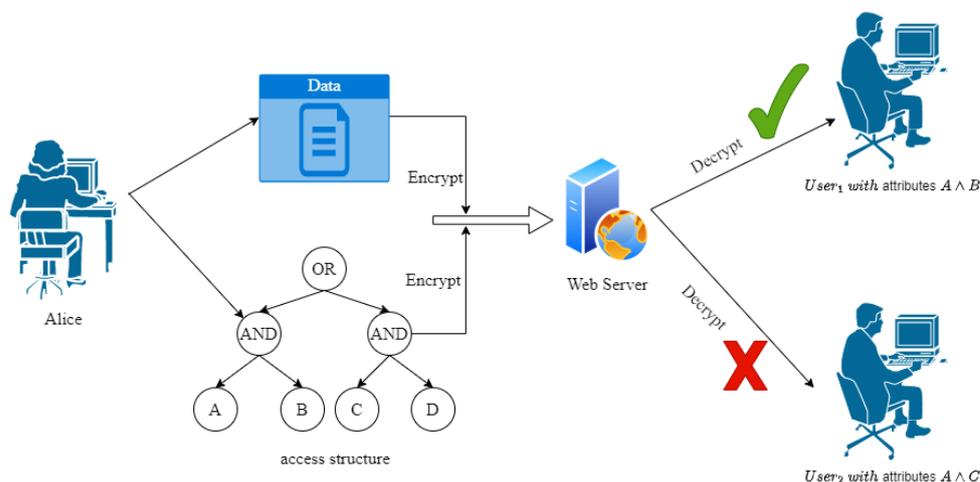


**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Currently, data have become something within people's reach, and more and more people are becoming aware of the ownership and usage of their data. What is "data confirmation"? The purpose of data confirmation is to legally establish ownership of the data and the right of the data owner to determine who can have access to the data. Data confirmation requires determining the type of rights, how they will be acquired, and how they will be distributed. With the popularity of cloud computing, people have begun to share data. There are many ways to share data, such as uploading to third-party trading platforms, cloud servers, Github [1], etc., resulting in the inability to ensure the privacy of users. Many scholars have begun to study data sharing, privacy issues [2,3], and how data are uploaded. In addition, the speed of data dissemination is extremely fast. Meng et al. [4] modeled network public opinion data to predict public opinion crisis warnings. Cao et al. [5] proposed a more comprehensive recommendation scheme based on real-world shared mobile data. The proposal of ciphertext-policy attribute-based encryption provides a good answer to these problems. CP-ABE allows the data owner to specify that only those who conform to the access policy can access the data. Figure 1 shows the execution process of CP-ABE. The data owner Alice has some data (*defined in the figure as "Data"*), and she sets a set of access policies  $((A \wedge B) \vee (C \wedge D))$  in the figure) according to the potential user subjectively, and combines *Data* with access policies to encrypt and upload to the server. At this time, there are two users  $User_1$  and  $User_2$  in the system;  $User_1$  has attributes *A* and *B*, and  $User_2$  has attributes *A* and *C*, then according to the access policies in the ciphertext,  $User_1$  can decrypt and access the data, but  $User_2$  fails to decrypt. However, due to the replicability of the data, ownership of the data cannot be determined. The development of blockchain [6–8] has brought the possibility of data rights confirmation. Due to its immutability and traceability, the information of the data owner cannot be tampered with

once it is on the chain, and it can be easily traced back to the source and destination of the data. However, as a result of the decentralized and public nature of blockchain technology, the privacy of the data owner cannot be guaranteed. Using traditional third-party hosting or issuing certificates of ownership does not guarantee that the data owner's source data will not be leaked. Therefore, there is the need of a scheme that solves the above problems.



**Figure 1.** The principle of encryption and decryption of CP-ABE.

Consider the following scenario: Alice wants to store her data on the cloud and share them with others, but she only wants a specific group of people to access the data. Therefore, she specifies an access policy and encrypts the data using CP-ABE before uploading it. Bob is a member of Alice's designated group, and he retrieves and decrypts the data from the cloud. Smith is Bob's friend, but he is not a part of the designated group. Smith contacts Bob and obtains a copy of the data. One day, Alice discovers that Smith is using her data and wants to seek compensation. However, Smith claims that the data are his own. How can Alice prove that the data belong to her?

In 2005, Sahai and Waters [9] introduced the notion of fuzzy identity-based encryption, which was later extended to attribute-based encryption (ABE). In an ABE system, the ciphertext and key are associated with the attribute set and access structure, and decryption only succeeds when the attribute set satisfies the access structure. Goyal et al. [10] suggested correlating the access policy with the ciphertext and key, respectively, and divided ABE for the first time into ciphertext-policy attribute-based encryption (CP-ABE) and key-policy attribute-based encryption (KP-ABE) in 2006. Bethencourt et al. [11] introduced the first CP-ABE system in 2007, embedding the access tree structure within ciphertext; however, it is challenging to deploy in practice. Waters et al. [12] built on Bethencourt et al.'s work in the following year and proposed a CP-ABE system with an efficient general access structure while also proving selection security under the standard model. In 2012, Lewko and Waters [13] developed a broad strategy for converting the standard model's concept of selection security into adaptive security. In today's cloud computing, CP-ABE has a significant influence. In 2015, Ning et al. [14] proposed a traceable and auditable CP-ABE scheme in cloud computing to address key abuse by dishonest users in the cloud storage environment [15], but it does not provide key revocation. Yu et al. [16] proposed a traceable and undeniable CP-ABE scheme based on Ning's work to solve the problem of semi-honest institutions illegally selling keys.

Under the big data environment [17], data can be used to verify the validity of the protocol [18] and can also be used to train the robot [19]. However, these web data have no real ownership. Yun Peng et al. [20] investigated the basic challenges surrounding data confirmation in 2016. Bing Guo et al. [21] presented a service system to defend the property rights of personal data in 2017. Shuaiyu Wang et al. [22] suggested a large data correct confirmation technique based on blockchain technology in the same year, but the issue is

that the data source cannot be verified. In 2018, Hailong Wang and his colleagues [23] introduced a novel approach for verifying big data using blockchain and digital watermarking technology, but the authority agency can access the data owner's source data. Although this solution can be applied to the environment of cloud storage, due to the limitation of its form of plaintext confirmation, the privacy of users cannot be guaranteed. Zhao et al. [24] developed a smart contract-based big data property right confirmation system the following year. In 2021, Zhou et al. [25] proposed a data ownership confirmation scheme based on consortium blockchain in IoT environments [26], with a focus on controlling the flow of data. However, the scheme cannot be applied to one-to-many environments such as cloud storage. Professors Jintai Ding and Ke Tang from Tsinghua University announced their plans to develop an innovative solution for managing large-scale data transactions. Their approach involves leveraging cutting-edge cryptography techniques and advanced mechanisms for economic design to create a robust and effective system for processing and exchanging data. By combining these two technologies, they aim to address the unique challenges associated with managing and securing large volumes of data, ultimately providing a reliable and efficient solution for businesses and organizations worldwide. This technique assures data transaction security while also increasing transaction efficiency. In 2022, Liu et al. [27] proposed a data ownership confirmation scheme based on the Ethereum blockchain and smart contracts. The parties authenticate their identities through a protocol for generating data fingerprints based on smart contracts. However, the article did not address the issue of user privacy protection on the public blockchain.

Based on the research status above, we propose a new data confirmation scheme in the cloud storage environment, focusing on user privacy protection and preventing the leakage of original plaintext data. The scheme can effectively protect the privacy of data owners while ensuring data confirmation, and in the process of confirmation, no one can access plaintext, thus reducing the risk of data leakage. We embed the data owner's identification information into CP-ABE using Paillier encryption and change the plaintext confirmation form to the ciphertext confirmation form. An audit phase is introduced at the end of the confirmation process.

Our contributions are as follows:

- (1) User privacy protection. We propose a new data confirmation scheme based on CP-ABE in the cloud storage environment. Users only need to embed the information with their own identity into the ciphertext after Paillier encryption and upload it to the cloud. They do not need to worry about revealing their identity.
- (2) Prevent original plaintext data leakage. During the entire right confirmation process, the authority  $AT$  can only access the ciphertext and only needs to process the ciphertext. This greatly reduces the risk of plaintext data leakage during the right confirmation process.
- (3) The scheme is safe and efficient. We reduce the scheme to the three-prime subgroup decision problem and prove that the scheme is safe, and through experimental analysis, our scheme is almost as efficient as the scheme proposed by Allison et al. [21] in terms of system setup, key generation, encryption, and decryption algorithms. Table 1 shows the comparison between our scheme and other data confirmation schemes.

Section 2 will present a formal definition and explanation of several fundamental concepts. Section 3 will focus on constructing the scheme, which will include defining the security requirements, implementing the scheme, and providing a security proof. In Section 4, we will conduct experiments and analysis to evaluate the effectiveness of the scheme. Finally, Section 5 will summarize the scheme and its contributions.

**Table 1.** Comparison of our scheme and other schemes.

	Zhou et al. [25]	Liu et al. [27]
Ways of identifying	Key verification	Fingerprint tracking protocol
Confirmation method	Consortium blockchain and smart contracts	Smart contract
Security assumption	Collision-resistant properties of hash function	null
Source data security	✓	✓
Can be applied to the cloud storage environment	×	×
	Wang et al. [23]	ours
Ways of identifying	Digital watermark	Pailler decryption
Confirmation method	Digital watermarking + blockchain	CP-ABE and Paillier encryption
Security assumption	CDH assumption	Subgroup decision problem for 3 primes
Source data security	×	✓
Can be applied to the cloud storage environment	✓	✓

**2. Preliminaries**

2.1. Access Structure

**Definition 1 ([9]).** Consider a set  $S$  containing  $n$  attributes, where each attribute is denoted by  $s_i$  for  $1 \leq i \leq n$ , a set  $\mathbb{A} \in 2^{\{s_1, \dots, s_n\}} \setminus \{\emptyset\}$  is an access structure on  $S$ , for  $\forall B, C \in \mathbb{A}$ : if  $B \in \mathbb{A}$  and  $B \subseteq C$ , then  $C \in \mathbb{A}$ , and  $\mathbb{A}$  is called monotonic. If a set is in  $\mathbb{A}$ , then it is called an authorized set, otherwise it is called a non-authorized set.

2.2. Linear Secret-Sharing Schemes

**Definition 2.** LSSS [28,29]. Suppose  $S$  represents the set of attributes, and let  $p$  be a prime number. A secret sharing scheme is denoted as  $\Pi$  and is operating on  $\mathbb{Z}_p$ . If the following two criteria are met by  $\Pi$ , then  $\Pi$  is referred to as linear:

1. The secret  $s \in \mathbb{Z}_p$  shared by each participant forms a column vector on  $\mathbb{Z}_p$ .
2. A secret sharing scheme  $\Pi$  has a shared generator matrix  $M$ , which is an  $l$ -by- $n$  matrix for every access structure  $\mathbb{A}$  defined on  $S$ . For  $i = 1, \dots, l$ , the  $i$ th line of  $M$  is marked as an attribute  $\rho(i)$  ( $\rho$  is a map that maps each row of matrix  $M$  to  $\Pi$ ). Given a vector  $\mathbf{v} = (s, r_2, \dots, r_n)$ , where  $s$  is the shared secret,  $r_2, \dots, r_n$  are randomly selected;  $\lambda = M_{l \times n} \cdot \mathbf{v}$  identifies the  $l$  shares of  $\Pi$  to the secret number  $s$ . Line  $i$  belongs to attribute  $\rho(i)$ .

**Secret Recovery:** Assuming  $\Pi$  accesses the LSSS of the structure  $\mathbb{A}$ ,  $S$  is the set of authorization attributes owned by the user, and  $M$  is the shared generation matrix. Define  $J = 1, \dots, j$  and  $J = \{j \mid \rho(j) \in S\}$ . For the vector  $\{\lambda_j\}_{j \in J}$  generated by the product of matrix  $M$  and secret vector  $\mathbf{v}$ , there exists a vector  $\mathbf{w} = \{w_j\}$  of integers in  $\mathbb{Z}_p$  such that  $\sum_{j \in J} M_j \cdot w_j = (1, 0, \dots, 0)$  and  $\rho(j)^T \cdot \mathbf{w} = s \pmod{N}$ ,  $\{w_j\}$  can be found in polynomial time, and  $\{w_j\}$  does not exist for non-authorized sets.

**Definition 3 ([30]).** Suppose  $\mathbb{A}$  is a monotonic access structure. In such a case, the definition of the shared generator matrix  $M$  yields the following conclusions:

- If  $M \in \mathbb{A}$ , there exists a vector  $\{\kappa_i\}$  of integers in  $\mathbb{Z}_p$  such that  $M^T \cdot \{\kappa_i\} = (1, 0, \dots, 0)^T$ .
- If  $M \notin \mathbb{A}$ , there exists a vector  $\{v_i\}$  of integers in  $\mathbb{Z}_p$  such that  $M^T \cdot \{v_i\} = (0, 0, \dots, 0)^T$  and  $v_1 = 1$ .

### 2.3. Composite Order Bilinear Groups

Prime order bilinear groups and composite order bilinear groups [31] are comparable; the difference is that the order of  $G_1, G_2, G_T$  is a composite number  $N$ , where  $N$  is the product of some large prime numbers, such as  $N = p_1 p_2 \cdots p_n$ , and  $e$  is a bilinear map,  $e : G_1 \times G_2 \rightarrow G_T$ . For any element  $a_i$  in  $G_{p_i}$  and element  $b_j$  in  $G_{p_j} (i \neq j)$ ,  $e(a_i, b_j) = 1$ .

**Assumption 1. (Subgroup decision problem for 3 primes):** We define the distribution shown below for a group generator  $\mathcal{G}$ :

$$\mathcal{G} \rightarrow \mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e)$$

$$g_1 \leftarrow G_{p_1}, E_3 \leftarrow G_{p_3}$$

$$\text{Distr} = (E_3, g, \mathbb{G})$$

$$X_1 \leftarrow G_{p_1 p_2}, X_2 \leftarrow G_{p_1}$$

In breaking Assumption 1, Algorithm  $\mathcal{A}$  has the following advantage:

$$\text{Adver1}_{\mathcal{G}, \mathcal{A}}(1^\lambda) := | \Pr[\mathcal{A}(\text{Distr}, X_1) = 1] - \Pr[\mathcal{A}(\text{Distr}, X_2) = 1] |$$

**Definition 4.** If  $\text{Adver1}_{\mathcal{G}, \mathcal{A}}(1^\lambda)$  is a negligible function of  $1^\lambda$  for every polynomial time algorithm  $\mathcal{A}$ , we claim that  $\mathcal{G}$  satisfies Assumption 1.

**Assumption 2.** We define the following distribution for a group generator  $\mathcal{G}$ :

$$\mathcal{G} \rightarrow \mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e)$$

$$g_1, E_1 \leftarrow G_{p_1}, E_2, F_2 \leftarrow G_{p_2}, E_3, F_3 \leftarrow G_{p_3}$$

$$\text{Distr} = (\mathbb{G}, g, E_1 E_2, F_3, E_2 F_3)$$

$$X_1 \leftarrow G, X_2 \leftarrow G_{p_1 p_3}$$

In breaking Assumption 2, Algorithm  $\mathcal{A}$  has the following advantage:

$$\text{Adver2}_{\mathcal{G}, \mathcal{A}}(1^\lambda) := | \Pr[\mathcal{A}(\text{Distr}, X_1) = 1] - \Pr[\mathcal{A}(\text{Distr}, X_2) = 1] |$$

**Definition 5.** If  $\text{Adver2}_{\mathcal{G}, \mathcal{A}}(1^\lambda)$  is a negligible function of  $1^\lambda$  for every polynomial time algorithm  $\mathcal{A}$ , we claim that  $\mathcal{G}$  satisfies Assumption 2.

**Assumption 3.** We define the following distribution for a group generator  $\mathcal{G}$ :

$$\mathcal{G} \rightarrow \mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e)$$

$$\gamma, t \leftarrow \mathbb{Z}_N$$

$$g_1 \leftarrow G_{p_1}, E_2, F_2, H_2 \leftarrow G_{p_2}, E_3, F_3 \leftarrow G_{p_3}$$

$$\text{Distr} = (\mathbb{G}, g, g^\gamma E_2, E_3, g^t F_2, H_2)$$

$$X_1 \leftarrow e(g, g)^{\gamma t}, X_2 \leftarrow G_{p_1 p_3}$$

In breaking Assumption 3, Algorithm  $\mathcal{A}$  has the following advantage:

$$\text{Adver3}_{\mathcal{G}, \mathcal{A}}(1^\lambda) := | \Pr[\mathcal{A}(\text{Distr}, X_1) = 1] - \Pr[\mathcal{A}(\text{Distr}, X_2) = 1] |$$

**Definition 6.** If  $\text{Adver3}_{\mathcal{G}, \mathcal{A}}(1^\lambda)$  is a negligible function of  $1^\lambda$  for every polynomial time algorithm  $\mathcal{A}$ , we claim that  $\mathcal{G}$  satisfies Assumption 3.

### 2.4. CDH Assumption

Computational Diffie–Hellman Assumption can be defined as follows:

Consider a finite cyclic group  $\mathbb{G}$  with  $n$  elements. The CDH assumption holds in  $\mathbb{G}$  if, given  $g, g^a$ , and  $g^b, g^{ab}$  cannot be calculated. The formal definition is as follows:

$$g, g^a, g^b \not\Rightarrow g^{ab}$$

for all efficient algorithms  $A$ :

$$Pr[A(g, g^a, g^b)] < \epsilon$$

where  $g$  is a generator in the group  $\mathbb{G}$ ,  $a, b \leftarrow \mathbb{Z}_n$ .

### 2.5. Paillier Encryption

The Paillier encryption algorithm [32] is a public key encryption algorithm based on the composite residual difficulty problem, which satisfies the additive homomorphic operation. It contains the following steps:

1. Key generation:

- (1) Obtain two large prime numbers  $p_1$  and  $p_2$  that satisfy  $gcd(p_1 p_2, (p_1 - 1)(p_2 - 1)) = 1$ . This ensures that the prime numbers  $p_1$  and  $p_2$  have equal lengths.
- (2) The following values are computed:  $n = p_1 p_2, \lambda = lcm((p_1 - 1), (p_2 - 1))$ .
- (3) Define  $\mathbb{L}(x) = (x - 1)/n$ .
- (4) Randomly select a positive integer  $g$  less than  $n^2$ , and there exists  $\mu = (\mathbb{L}(g^\lambda \bmod n^2))^{-1} \bmod n$ .
- (5) System public key  $pk = (n, g)$ , and system secret key  $sk = (\lambda, \mu)$ .

2. Encryption:

Given the plaintext  $\mathbb{M}$ , randomly select  $\gamma \in \mathbb{Z}_{n^2}^*$  and calculate  $\mathbb{C} = g^{\mathbb{M}} \gamma^n \bmod n^2$ .

3. Decryption:

$\mathbb{M} = \mathbb{L}(\mathbb{C}^\lambda \bmod n^2) \times \mu \bmod n$ .

### 2.6. Fully Secure CP-ABE

Allison et al. [33] proposed a CP-ABE scheme that is fully secure. The scheme is built using composite order bilinear groups and *LSSS*. Four algorithms can be executed in polynomial time:

- $Setup(\varphi, \mathcal{U}) \rightarrow PK, MSK$ : The setup procedure receives two input parameters: the security parameter  $\varphi$ , which determines the level of security required, and the attribute universe  $\mathcal{U}$ , which defines the set of attributes. It then generates two output values: the public parameter  $PK$ , which can be shared publicly and used for encryption and decryption, and the master key  $MSK$ , which is kept secret and used for key generation.
- $KeyGen(MSK, \mathcal{S}, PK) \rightarrow SK$ : Given the master key  $MSK$ , the user’s attribute  $\mathcal{S}$ , and the public parameter  $PK$  as input, the key generation algorithm computes the decryption key  $SK$  as its output. The decryption key  $SK$  can be used to decrypt data encrypted using the corresponding attribute  $\mathcal{S}$  and the public parameter  $PK$ .
- $Encrypt((A, \rho), PK, \mathcal{M}) \rightarrow CT$ : To encrypt a plaintext  $\mathcal{M}$ , the encryption algorithm takes as input a matrix  $A$ , where each row of the matrix is mapped to an attribute  $\rho$ , along with the public parameter  $PK$ . The encryption algorithm computes the ciphertext  $CT$  as its output.
- $Decrypt(CT, PK, SK) \rightarrow \mathcal{M}$ : Given the ciphertext  $CT$ , the public parameter  $PK$ , and the decryption key  $SK$ , the decryption algorithm computes the corresponding plaintext  $\mathcal{M}$  as its output. The decryption key  $SK$  must be associated with an authorization set mapped to rows of the matrix used during encryption, otherwise the decryption will fail.

### 3. Construction

#### 3.1. Membership

Our system consists of five parties (as shown in Figure 2). The data owner (*Do*) is in charge of data encryption and uploading. The data user (*Du*) is responsible for retrieving and decrypting the data submitted by the data owner from the cloud. The authority (*AT*) is in charge of giving decryption keys to data users, participating in the ciphertext’s signature, and storing the credentials of the data owner. The public auditor (*PA*) is in charge of publicly auditing the ciphertext and extracting the information of the ciphertext owner from credentials. Finally, the cloud server (*Cloud*) is responsible for storing the ciphertext uploaded by the data owner.

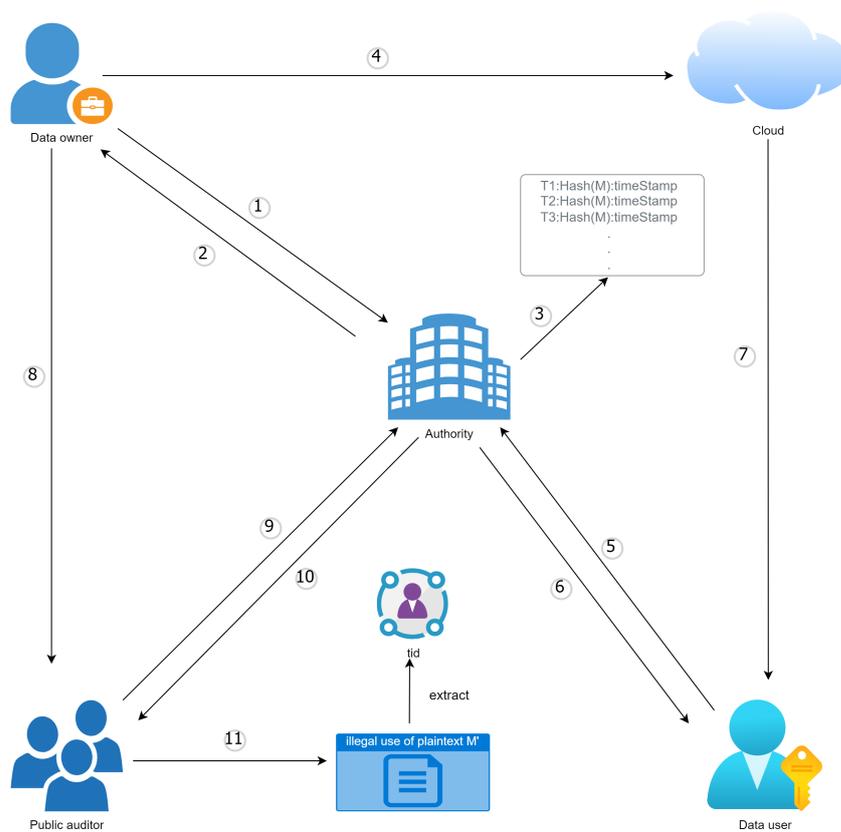


Figure 2. Membership.

#### 3.2. Security

##### 3.2.1. IND-CPA Security

We can rephrase the description of the *IND – CPA* security game process for our proposed scheme, which is equivalent to the one proposed by Allison et al. [21], as follows:

- *Setup:* The adversary  $\mathcal{A}$  is given the public parameter  $PK$  after the challenger  $\mathcal{B}$  calls the  $Setup(1^\varphi, U)$  algorithm.
- *Phase 1:* Adversary  $\mathcal{A}$  can dynamically request the decryption keys  $Sk_i$  associated with attribute sets  $S_1, \dots, S_{q_r}$  from the challenger  $\mathcal{B}$ . In response,  $\mathcal{B}$  executes the key generation algorithm to generate  $Sk_i$  and sends it to  $\mathcal{A}$ .
- *Challenge:* Adversary  $\mathcal{A}$  provides two equal-length messages  $M_1$  and  $M_2$  and a generator matrix  $A^*$  that corresponds to an access structure  $\mathbb{A}^*$  that does not satisfy  $S_1, \dots, S_{q_r}$  to the challenger  $\mathcal{B}$ . Then  $\mathcal{B}$  randomly chooses a bit  $\sigma \in \{0, 1\}$  and generates the ciphertext  $CT_{A^*, T}$  by calling the encryption algorithm with  $sk_{D_o}, \langle A^*, \rho \rangle, PK$ , and  $M_\sigma$ . Finally,  $\mathcal{B}$  sends  $CT_{A^*, T}$  to adversary  $\mathcal{A}$ .

- Phase 2: Adversary  $\mathcal{A}$  keeps asking  $\mathcal{B}$  for decryption keys  $Sk_i$  corresponding to attribute sets  $S_{q_{r+1}}, \dots, S_{q_r}$ , where each set cannot satisfy the access structure  $\mathbb{A}^*$ . Upon each request,  $\mathcal{B}$  calls the key generation algorithm and sends  $Sk_i$  to adversary  $\mathcal{A}$ .
- Guess:  $\mathcal{A}$  outputs a guess  $\sigma' \in \{0, 1\}$ .

The advantage of the adversary in this game is defined as:

$$Adv_{(\mathcal{A})} = | Pr[\sigma' = \sigma] - 1/2 |$$

**Definition 7.** If we assume that any adversary with polynomial time has only a negligible advantage in winning the aforementioned game, we can confidently assert that our scheme is secure.

### 3.2.2. Dishonest User Game (Non-Replicability of Ciphertext)

The dishonest user game of this scheme is defined as follows: A user attempts to confuse the auditor by forging the authority’s signature and republishing a ciphertext. The game is played by a challenger and an adversary.

-Setup: Challenger  $\mathcal{B}$  starts the  $Setup(1^q, U)$  algorithm and sends the public parameters  $PK$  and  $sk_{D_0}$  to the attacker  $\mathcal{A}$ .

Ciphertext Generation: Challenger  $\mathcal{B}$  generates the ciphertext  $CT_{A,T}$  through the  $Encrypt$  algorithm and sends it to  $\mathcal{A}$ ;  $\mathcal{A}$  generates a new ciphertext  $CT'_{A,T'}$  according to the initial ciphertext  $CT_{A,T}$ .

Output:

If  $Decrypt(Sk_{Du}, CT_{A,T}) = Decrypt(Sk_{Du}, CT'_{A,T'})$  and  $C'_0 = M \cdot e(g^{\alpha\beta}, g^{sT'})$  then we say that the attacker successfully copies the ciphertext.

The adversary’s advantage in the dishonest user game is defined as

$$Adv = | Pr[\mathcal{A} \text{ success}] |$$

**Definition 8.** If the probability of a polynomial-time adversary winning the game described above is negligible, then we consider the ciphertext of our scheme to be secure and irreproducible.

In order to satisfy the requirement of data confirmation, the conventional CP-ABE scheme is insufficient. To ensure auditing capabilities in our CP-ABE scheme, we have developed a method that involves incorporating the data owner’s unique identifier (such as an address or ID number) into the ciphertext using Paillier encryption. The process of our scheme is illustrated in Figure 3.

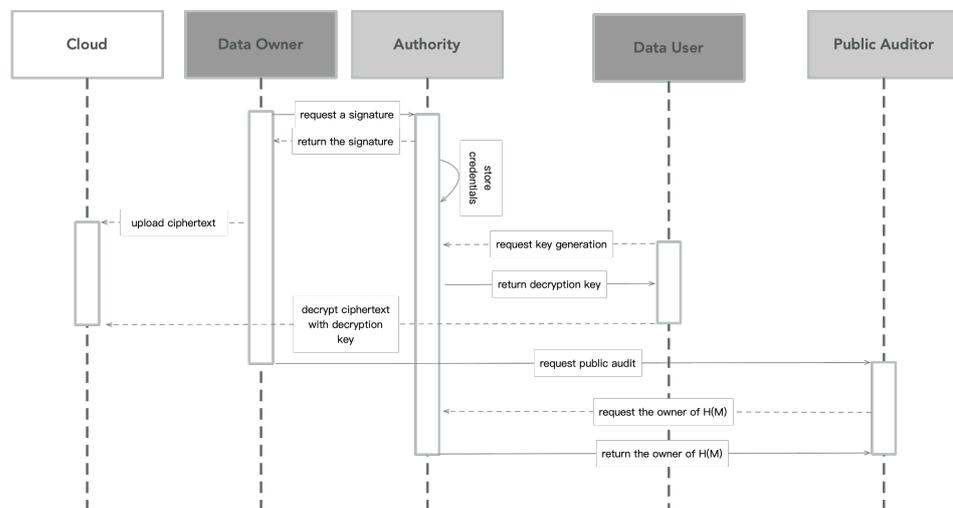


Figure 3. Process.

### 3.3. Implementation

- Setup**  $(1^\varphi, U) \rightarrow PK, MSK, Pk_{AT}, Sk_{AT}$ : In the setup phase of our system, we provide the security parameter  $\varphi$  and the user attribute universe  $U$  as inputs to the setup algorithm. This algorithm then generates a group  $G$  of order  $N = p_1 p_2 p_3$ , a mapping  $e$ , an integer group  $\mathbb{Z}_N$ , and a hash function  $H : H(x) \rightarrow \mathbb{Z}_N$ . This setup process establishes the necessary parameters and functions to enable secure and efficient cryptographic operations in our system. The resulting setup allows us to implement our system in a manner that satisfies our security and performance requirements. Then the system proceeds to select random parameters  $\alpha, a \in \mathbb{Z}_N$ , and the generator  $g \in G_{p_1}$ . For each attribute  $s \in U$ , the system randomly selects a corresponding value  $u_i \in \mathbb{Z}_N$ . The system global parameter is set as

$$PK = (N, G, \mathbb{Z}_N, H(x), g, g^a, e(g, g)^\alpha, \{U = g^{u_i}\}_{\forall i \in U})$$

$MSK = (\alpha, g_3)$  ( $g_3 \in G_{p_3}$  and is a generator) and  $MSK$  is sent to the authority  $AT$ ;  $AT$  performs the following steps locally: randomly selecting two safe large primes  $p$  and  $q$ , which satisfy  $\gcd(pq, (p-1)(q-1)) = 1$ , calculating  $n = pq, \lambda = \text{lcm}(p-1, q-1)$ , and then randomly selecting a positive integer  $g_1$  that is less than  $n^2$ . Next,  $AT$  computes  $\mu = (L(g^\lambda \text{mod} n^2))^{-1} \text{mod} n$  and randomly selects a value  $\beta \in \mathbb{Z}_N$ . The public parameter  $Pk_{AT} = (n, g_1, g^\beta)$  is generated, whereas the private key  $Sk_{AT} = (\lambda, \mu, \beta)$  is stored locally.

- Encrypt**  $(Pk_{AT}, \langle A, \rho \rangle, PK, M) \rightarrow CT_{A,T}$ :

**Step 1:** The unique identifier (e.g., ID number, address, mailbox, etc.) is hashed by data owner  $Do$  and mapped to an integer in  $\mathbb{Z}_N$ , denoted as

$$t_{id} = H(\text{identity}) \rightarrow \mathbb{Z}_N$$

After mapping the data owner's unique identifier to an integer in  $\mathbb{Z}_N$ ,  $Do$  chooses a value  $r \xleftarrow{R} \mathbb{Z}_{n^2}^*$  and employs Paillier encryption to generate the encrypted output  $T = g_1^{t_{id}} r^n \text{mod} n^2$ . The Algorithm 1 is as follows (here we assume the unique identifier string is "address"):

---

#### Algorithm 1 Encrypt $t_{id}$

---

**Input:** String "address"

- $addrHash \leftarrow$  Convert "address" to a byte array after hashing;
- $t_{id} \leftarrow$  Map  $addrHash$  into  $\mathbb{Z}_N$ ;
- $r \leftarrow$  Randomly pick an element from  $\mathbb{Z}_N$ ;
- $T \leftarrow$  Use Paillier encryption to obtain encrypted identity information;

**Output:**

$$t_{id}=79847630022358710946125273965671104052858\ 065717629025639108307113838327353$$


---

**Step 2:** To encode the access structure for the data, the owner of the data,  $Do$ , creates a shared generator matrix  $A$  with dimensions  $l$  by  $n$  using the LSSS. First, a secret number  $s \xleftarrow{R} \mathbb{Z}_n$  is randomly selected. Then,  $n - 1$  random numbers  $y_2, \dots, y_n \xleftarrow{R} \mathbb{Z}_n$  are selected to generate a vector  $\mathbf{y} = (s, y_2, \dots, y_n)$ . Finally, random numbers  $r_i \xleftarrow{R} \mathbb{Z}_N$  are chosen for each row  $A_{i \in [l]}$  of matrix  $A([l])$  that represents the entire set of  $\{1, 2, \dots, l\}$ ,  $H(M)$  is obtained by taking a hash of the plaintext  $M$  and mapping it to  $G_T$  to generate ciphertext:

$$\overline{CT_{A,T}} = \langle \overline{C_0} = G_T(H(M)) \cdot e(g, g)^{as}, C_1 = g^{sT}, C_{i,1} = g^{aTA_i \cdot \mathbf{v}} U_{\rho(i)}^{-r_i}, C_{i,2} = g^{r_i}, C_{i,3} = g^{\beta TA_i \cdot \mathbf{v}} \rangle$$

**Step 3:** Both  $\overline{C}_0, g^s$  and  $T$  are sent to the authority  $AT$  for decryption. The decryption process begins with  $AT$  decrypting  $G_T(H(M))$  using the following method:

$$G_T(H(M)) = \overline{C}_0 / e(g^\alpha, g^s)$$

After successfully decrypting  $G_T(H(M))$ , the authority  $AT$  checks if it already has a record of  $G_T(H(M))$  in its database. If a record already exists, the application is rejected; otherwise,  $AT$  utilizes their private key  $\beta$  to sign the message and generates

$$C'_0 = G_T(H(M)) \cdot e(g, g)^{\alpha\beta}$$

and stores the data credentials of  $Do$  in the local database in the form of  $T : G_T(H(M)) : timeStamp$ . By following this process, it is guaranteed that there is only one legitimate owner associated with the original data source. This measure also serves as a safeguard against any attempts by malicious actors to produce ciphertext and assert false ownership over the data. Furthermore, this also serves to prevent  $AT$  from directly accessing the plaintext, which enhances the security of the system. Finally,  $C'_0$  is sent back to the data owner  $Do$  for further processing. The user credentials setting Algorithm 2 is as follows:

---

**Algorithm 2** Store user credentials

---

**Input:**  $\overline{C}_0, g^s, T$

- 1: Divide  $\overline{C}_0$  by  $e(g^\alpha, g^s)$  to get  $G_T(H(M))$ ;
  - 2: **if** Retrieving  $G_T(H(M))$  locally is empty **then**
  - 3:   Element  $P = e(g, g)^{\alpha\beta}$ ;
  - 4:   Date  $date =$  Get the current time through the time function;
  - 5:    $Recordlist[] \leftarrow (T, G_T(H(M)), date)$ ;
  - 6: **end if**
  - 7: **return**  $P * G_T(H(M))$ ;
- 

**Step 4:**  $Do$  first calculates

$$C_0 = M \cdot (C'_0 / G_T(H(M)))^T$$

after receiving  $C'_0$ , afterwards, the ciphertext is assigned the value

$$CT_{A,T} = \langle C_0 = M \cdot e(g, g)^{\alpha s T \beta}, C_1 = g^{sT}, C_{i,1} = g^{aTA_i \cdot v} \mathcal{U}_{\rho(i)}^{-r_i}, C_{i,2} = g^{r_i}, C_{i,3} = g^{\beta TA_i \cdot v} \rangle$$

and uploads  $CT_{A,T}$  to the cloud.

**Note:** A notable characteristic of this scheme is the possibility of having multiple owners for a given *data*, which is made feasible by the additive homomorphism property of Paillier encryption. For example, in a scenario where the data are jointly owned by two parties, denoted as  $Do_1$  and  $Do_2$ , they can both hash their unique identifiers and use them to generate separate *Paillier* ciphertexts  $T_1$  and  $T_2$  using different random numbers, then  $Do_1, Do_2$  calculate  $T_1 = g_1^{t_{id_1}} r_2^n \bmod n^2, T_2 = g_1^{t_{id_2}} r_2^n \bmod n^2$ , let  $T = T_1 \cdot T_2$ .

During the entire encryption stage, we have realized data confirmation. Hash the plaintext and map it to  $G_T$  for encryption ( $\overline{C}_0 = G_T(H(M)) \cdot e(g, g)^{\alpha s}$ ) and send it to  $AT$ ;  $AT$  only needs to perform division and signature operations on  $\overline{C}_0$ , and store user ID  $T$  locally as a certificate. Therefore,  $AT$  cannot touch the plaintext.

3. **KeyGen** ( $MSK, S, PK$ )  $\rightarrow Sk_{Du}$ : The generation of the decryption key in this scheme is a collaborative process between  $Du$  and  $AT$ ;  $Du$  first chooses a random number  $t \xleftarrow{R} \mathbb{Z}_N$  as a parameter. Next,  $Du$  forwards their personal set of attributes  $S$  and

the value  $g^t$  to "AT as part of its request to generate a key. Then, AT selects random numbers  $h \xleftarrow{R} \mathbb{Z}_N$  and  $R_0, R_1, R_2, R_3, R_i \in G_{p_3}$  to generate part of the decryption key

$$Sk_{pri} = \langle S, \bar{D} = g^{\beta\alpha}, \bar{D}_1 = g^{ah}, \bar{D}_2 = g^{\beta h}, \bar{D}_3 = g^{th}, \{\bar{D}_i = \mathcal{U}_i^h\}_{i \in S} \rangle$$

Finally, AT transmits the decryption key  $Sk_{pri}$  and a collection of values labeled as  $\{R_0, R_1, R_2, R_3, R_i\}$  to Du, and Du generates the decryption key locally using these values:

$$Sk_{Du} = \langle S, D = \bar{D}R_0, D_1 = \bar{D}_1^t R_1, D_2 = \bar{D}_2^t R_2, D_3 = \bar{D}_3 R_3, \{D_i = \bar{D}_i^t R_i\}_{i \in S} \rangle$$

4. **Decrypt** ( $Sk_{Du}, CT_{A,T}$ )  $\rightarrow M$  : The decryption key allows Du to decrypt the ciphertext and obtain access to the data. The decryption algorithm searches for a vector  $\mathbf{w}$  such that  $A_i^T \cdot \mathbf{w} = (1, 0, \dots, 0)^T (i \in S)$ , if the attributes of Du do not satisfy the access policy, then there is only one vector  $\{\kappa_i\}$ , such that  $A_i^T \cdot \{\kappa_i\} = (0, 0, \dots, 0)^T (i \in S)$  and  $\kappa_1 = 1$ , the plaintext M is obtained by the following formula:

$$F = e(C_1, DD_1 D_2)$$

$$E = \prod_{\rho(i) \in S} (e(C_{i,1} C_{i,3}, D_3) e(C_{i,2}, D_i))^{w_i}$$

$$M = C/F/E$$

5. **Audit** ( $PK, M, M^*, Pk_{AT}, Sk_{AT}, MSK$ )  $\rightarrow t_{id}$ : If the data owner Do suspects that his data have been infringed upon or abused, he can prove his ownership by interacting with the public auditor PA and the authority AT. This interaction serves two purposes:
  - (a) To demonstrate that Do was the first to upload the data;
  - (b) To prove that the ciphertext corresponding to the data is indeed generated by Do.

**Step 1:** To prove that Do is the first to upload the data, the source data M and  $CT_{A,T}$  are sent by Do to the public auditor PA. PA obtains the hash value of the source data M by applying the hash function  $H(x)$  and sends it to the authority AT to identify the owner of the plaintext.

**Step 2:** First, PA carries out a comparison:

$$M \stackrel{?}{=} M^*, C_0 \stackrel{?}{=} M \cdot e(g^{\alpha\beta}, C_1)$$

If they are equal, PA enter the  $t_{id}$  extraction process using  $n, \lambda$ , defines  $L(x) = (x - 1)/n$ , calculates  $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ , then by  $L(T'^\lambda \bmod n^2) \times \mu \bmod n$  to extract the Do's  $t_{id}$ .

**Step 3:** PA is needed to verify whether the given equation is valid or false.

$$H(identity) \rightarrow \mathbb{Z}_N \stackrel{?}{=} t_{id}$$

Assuming the equation is satisfied, we can conclude that the data belong to the user Do. Let us take the unique identifier "address" during encryption as an example: the decryption Algorithm 3 and the decrypted  $t_{id}$  are as follows:

---

**Algorithm 3** Decrypt  $t_{id}$

---

- 1:  $\lambda \leftarrow (p - 1) * (q - 1)$ ;
- 2:  $\mu \leftarrow$  Get  $\mu$  according to the formula  $(L(g^\lambda \bmod n^2))^{-1} \bmod n$ ;
- 3:  $t_{id} \leftarrow$  Obtain the  $t_{id}$  in the ciphertext according to the decryption algorithm  $L(T'^\lambda \bmod n^2) \times \mu \bmod n$ ;

**Output:**

$$t'_{id} = 79847630022358710946125273965671104052858\ 065717629025639108307113838327353$$


---

If the data are generated by multiple users, then  $t_{id_1} + t_{id_2} \equiv L(g^{\lambda(t_{id_1} + t_{id_2})} \cdot (r_1 \cdot r_2)^{\lambda n}) \cdot \mu \pmod n$ , PA verifies  $H(identity_1) \rightarrow \mathbb{Z}_N + H(identity_2) \rightarrow \mathbb{Z}_N \stackrel{?}{=} t_{id_1} + t_{id_2}$ .

During the entire audit phase, PA needs to do two things:

- (1) Compare whether the leaked plaintext is the same as that owned by Do and calculate whether the ciphertext is generated by Do through the formula  $C_0 \stackrel{?}{=} M \cdot e(g^{\alpha\beta}, C_1)$ ;
- (2) Obtain the user credential  $T$  corresponding to the plaintext in the AT's database and obtain the owner of the plaintext through Paillier decryption.

### 3.4. Correctness

$$\begin{aligned}
 e(C_1, D) &= e(g^{sT}, g^{\beta\alpha} R_0 g^{\beta ht} R_2 g^{aht} R_1) = \\
 &e(g^{sT}, g^{\beta\alpha}) e(g^{sT}, g^{\beta ht}) e(g^{sT}, g^{aht}) \\
 \prod_{\rho(i) \in S} (e(C_{i,1} C_{i,3}, D_3) e(C_{i,2}, D_i))^{w_i} &= \\
 \prod_{\rho(i) \in S} (e(g^{aTA_i \cdot \mathbf{v}} \mathcal{U}_{\rho(i)}^{-r_i} g^{\beta TA_i \cdot \mathbf{v}}, g^{th} R'_i) \\
 &e(g^{r_i}, \mathcal{U}_i^{ht} R'_i)) \\
 &= (e(g^{aT}, g^{ht}) e(g^{\beta T}, g^{ht}))^{\sum A_i \cdot \mathbf{v} \cdot w_i} \\
 &= e(g^{aTs}, g^{th}) e(g^{\beta Ts}, g^{th}) \\
 F &= e(g^{sT}, g^{\beta\alpha}) e(g^{sT}, g^{\beta ht}) e(g^{sT}, g^{aht}) / \\
 &e(g^{aTs}, g^{th}) e(g^{\beta Ts}, g^{th}) \\
 &= e(g, g)^{\alpha\beta sT} \\
 C_0 / F &= M
 \end{aligned}$$

### 3.5. IND-CPA Security

Suppose there is an adversary  $\mathcal{A}$  who can eavesdrop on the channel between the user and the data owner, and he can obtain the ciphertext corresponding to the plaintext within a limited time, so as to crack the key and gain unlimited access to the ciphertext. Our scheme's IND – CPA security is analogous to the IND – CPA security of the CP – ABE scheme proposed by Allison and his colleagues in [33], and we only prove Assumption 1 here. To begin with, we create a semi-functional ciphertext (defined as SF-C) and a semi-functional key (defined as SF-K) in the following format:

**SF-C:** We define  $g_2$  as the generator element of the group  $G_{p_2}$ . It randomly selects  $f \xleftarrow{R} \mathbb{Z}_N$ , for each attribute, selects  $z_i \xleftarrow{R} \mathbb{Z}_N$ , then selects  $\gamma_i \xleftarrow{R} \mathbb{Z}_N$  for each row of the shared generator matrix and two random vectors  $\mathbf{u}, \mathbf{w} \in \mathbb{Z}_N^n$ , SF-C is defined as follows:  $C_1 = g^{sT} \cdot g_2^f, C_2 = g^{\beta sT} \cdot g_2^{\beta f}, C_{i,1} = g^{aTA_i \cdot \mathbf{v}} \mathcal{U}_{\rho(i)}^{-r_i} g_2^{A_i \cdot \mathbf{u} + \gamma_i z_{\rho(i)}}, C_{i,2} = g^{r_i} g_2^{-\gamma_i}, C_{i,3} = g^{\beta TA_i \cdot \mathbf{v}} \cdot g_2^{A_i \cdot \mathbf{w}}$

**SF-K:** We can create two types of SF-K by randomly selecting the parameters  $d, h, c \xleftarrow{R} \mathbb{Z}_N, R'_0, R'_1, R'_2, R'_3, R'_i \in G_{p_3}$  as follows:

**Type 1 :**  $D = g^{\beta\alpha} g_2^d R'_0, D_1 = g^{aht} g_2^d R'_1, D_2 = g^{\beta ht} g_2^d R'_2, D_3 = g^{ht} g_2^c R'_3, \{D_i = \mathcal{U}_i^{ht} R'_i g_2^{cz_i}\}$

**Type 2 :**  $D = g^{\beta\alpha} g_2^d R'_0, D_1 = g^{aht} g_2^d R'_1, D_2 = g^{\beta ht} g_2^d R'_2, D_3 = g^{ht} R'_3, \{D_i = \mathcal{U}_i^{ht} R'_i\}$  (let  $c = 0$ )

Upon decrypting an SF-C with an SF-K, an additional term is introduced into the plaintext due to the semi-functional properties of the key and ciphertext:

$$e(g_2, g_2)^{3fd - 2u_1c}$$

where  $u_1$  represents the first item of the vector  $\mathbf{u}$ . We will now introduce a series of games to analyze the security of our proposed scheme:

$Game_{Real}$ : In this game, both the ciphertext and the decryption key are valid, and the security of the scheme is not compromised.

$Game_0$ : We define a game where all keys are normal, but the challenge ciphertext is SF-C. Let  $q$  be the number of times the attacker requests the key. For  $k \in [1, q]$ , we define:

$Game_{k,1}$ : The challenge ciphertext is SF-C, the first  $k - 1$  keys requested by the adversary are SF-K of **Type 2**, the  $k$ th key is SF-K of **Type 1**, and the rest are normal.  $Game_{k,2}$ : We define a game where the challenge ciphertext is SF-C, the first  $k$  keys are SF-K of **Type 2**, and the remaining keys are normal.

At the end of the game, we play the game's last round ( $Game_{final}$ ): all the keys are **Type 2** SF-K, and the ciphertext is generated by semi-functionally encrypting random messages without using the two messages supplied by the adversary.

**Lemma 1.** Assume the existence of a polynomial-time algorithm  $\mathcal{A}$  such that  $Game_{Real} Adv_{\mathcal{A}} - Game_0 Adv_{\mathcal{A}} = \epsilon$ . where  $\epsilon$  is a non-negligible value. We can find a polynomial-time algorithm  $\mathcal{B}$  to break **Assumption 1** by  $\epsilon$ .

**Proof.** Sending  $g, g_3, X$  to  $\mathcal{B}$ ,  $\mathcal{B}$  will simulate  $Game_{Real}$  or  $Game_0$  with adversary  $\mathcal{A}$ ;  $\mathcal{B}$  randomly selects  $\alpha, a, \beta \in \mathbb{Z}_N$ , and selects a random exponent  $u_i \in \mathbb{Z}_N$  for each attribute in the system, then randomly selects two safe large prime numbers  $p, q$  such that  $gcd(pq, (p - 1)(q - 1)) = 1$ , calculates  $n = pq, \lambda = lcm(p - 1, q - 1)$ , then randomly selects a positive integer  $g_1$  less than  $n^2$ , and  $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ , the public parameter

$$PK = (N, g, g^a, g^\beta, e(g, g)^\alpha, \{U = g^{u_i}\}_{\forall i \in U})$$

and public key

$$Pk = (n, g_1, g^\beta)$$

are sent to  $\mathcal{A}$ .  $\square$

Next,  $\mathcal{A}$  sends two equal-length messages  $M_0, M_1, T$  generated by his own unique identity and a shared generator matrix  $(A^*, \rho)$  to  $\mathcal{B}$ ,  $\mathcal{B}$  implicitly sets  $g^{sT}$  to the part of  $G_{p_1}$  (and possibly  $G_{p_1 p_2}$  element). Then,  $\mathcal{B}$  flips a coin and pick  $\sigma \in \{0, 1\}$  and sets:

$$C_0 = M_\sigma e(g^{\alpha\beta}, X), C_1 = X$$

then randomly selects  $y'_2, \dots, y'_n, r'_i \xleftarrow{R} \mathbb{Z}_N$ , sets the vector  $\mathbf{v}' = (1, y'_2, \dots, y'_n)$ , then sets  $C_{i,1} = X^{aTA_i \cdot \mathbf{v}'} X^{-r'_i u_{\rho(i)}}$ ,  $C_{i,2} = X^{r'_i}$ ,  $C_{i,3} = X^{\beta TA_i \cdot \mathbf{v}'}$ . We implicitly set  $\mathbf{v}$  to  $(s, sy'_1, \dots, sy'_n)$ ,  $r_i = sr'_i$ , so when  $X \in G_{p_1}$ , it is a correctly distributed normal ciphertext.

If  $X \in G_{p_1 p_2}$ , let  $g_2^{f'}$  be the  $G_{p_2}$  part of  $X$  ( $X = g^s g_2^{f'}$ ), so  $C_1 = g^{sT} g_2^{f'T}$ ,  $C_2 = g^{s\beta T} g_2^{\beta f'T}$ ,  $C_{i,1} = g^{saTA_i \cdot \mathbf{v}'} g^{-sr'_i u_{\rho(i)}} \cdot g_2^{f'aTA_i \cdot \mathbf{v}' - f'r'_i u_{\rho(i)}}$ ,  $C_{i,2} = g^{sr'_i} g_2^{f'r'_i}$ ,  $C_{i,3} = g^{s\beta TA_i \cdot \mathbf{v}'} g_2^{\beta TA_i \cdot \mathbf{v}'}$ . Let

$$\mathbf{u} = f'aT\mathbf{v}', \gamma_i = -f'r'_i, z_{\rho(i)} = u_{\rho(i)}, \mathbf{w} = f'\beta T\mathbf{v}'$$

this is a correctly distributed semi-functional ciphertext. We simulated and ran local experiments to test our scheme against Choose Plaintext Attack, and in both  $X \in G_{p_1}$  and  $X \in G_{p_1 p_2}$  scenarios, attacker  $\mathcal{A}$  was unable to decrypt the data. Figure 4 illustrates the process of a chosen plaintext attack and the experimental results obtained by the attacker. Therefore,  $\mathcal{A}$  can break **Assumption 1** with the advantage of  $\epsilon$ .

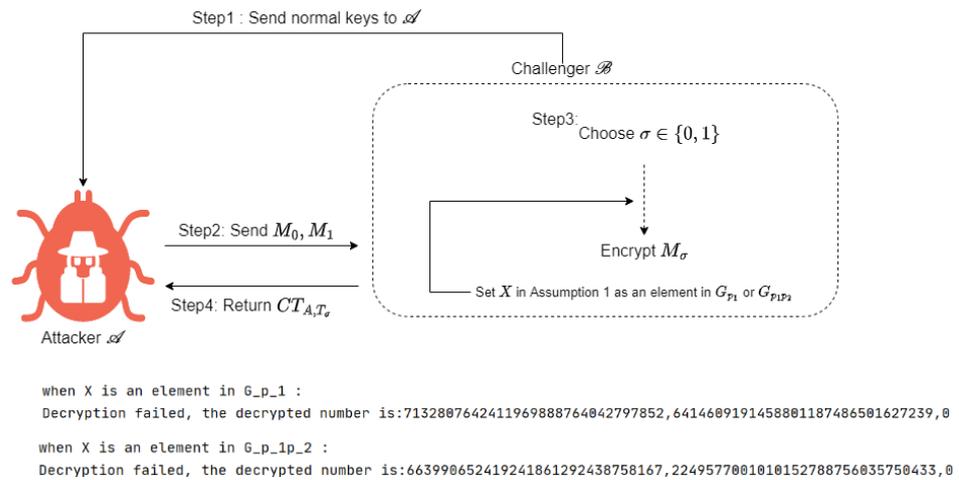


Figure 4. Attack process and output results.

Assumptions 2 and 3 can be proved by similar constructions above; see Allison’s scheme [33] for details.

### 3.6. Ciphertext Non-Replicability

Suppose there is an adversary  $\mathcal{A}$  who can eavesdrop on the channel between the data owner and  $AT$ . The purpose of  $\mathcal{A}$  in this game is to obtain the signature of  $AT$  and embed its own  $T'$  in the ciphertext and replace the identity of the data owner in the ciphertext data with its own identity, so as to obtain the ownership of the data. We assume that the adversary will not send his identity information to  $AT$  without being able to copy the ciphertext (even if sent, it does not pass authentication).

**Lemma 2.** Assume that there is a polynomial-time algorithm  $\mathcal{A}$  that can break the **CDH Assumption** with the advantage of  $\epsilon$  in the polynomial time, then we can construct a polynomial-time algorithm  $\mathcal{B}$  that falsifies ciphertext with the advantage of  $\epsilon$ .

**Proof.**  $\mathcal{B}$  first runs the  $Setup(1^\varphi, U) \rightarrow (PK, MSK, Pk_{AT}, Sk_{AT})$  algorithm, and  $PK, Pk_{AT}$  are sent to the adversary  $\mathcal{A}$ .  $\square$

**Ciphertext generation:**  $\mathcal{B}$  first interacts with  $AT$  to generate the ciphertext  $CT_{A,T}$ , and sends  $CT_{A,T}$  to  $\mathcal{A}$ . The adversary has two ways to generate its own ciphertext:

**Case 1:** After the adversary (dishonest user) decrypts the ciphertext and obtains the plaintext  $M$ , it regenerates the ciphertext  $CT'$  by itself. This method is obviously not advisable, because even if the original decryption key of the ciphertext is generated, it is unable to decrypt  $CT'$  and  $G_T(H(M))$  has been stored locally in the authority.

**Case 2:** The adversary obtains the signature and generates the ciphertext by eavesdropping on the channel between the data owner and  $AT$ , and sending information that is beneficial to  $\mathcal{A}$  to  $AT$ ;  $\mathcal{B}$  randomly selects  $s \xleftarrow{R} \mathbb{Z}_n$ , hashes the plaintext  $M$  and maps it to  $G_T, \overline{C}_0 = G_T(H(M)) \cdot e(g, g)^{as}$  and  $\overline{C}_0, g^s$  are sent to  $\mathcal{A}$ .

The adversary  $\mathcal{A}$  attempts to generate a random number  $s'$  such that  $e(g, g)^{as'} = e(g, g)^{as}$ , so he can send  $\overline{C}'_0 = G_T(H(M)) \cdot e(g, g)^{as'}, g^{s'}$  and his own identity  $T'$  to obtain the signature of  $AT$ , and then according to  $Encrypt(Pk_{AT}, \langle A, \rho \rangle, PK, M) \rightarrow CT_{A,T}$  algorithm to generate ciphertext and publishes it,  $\mathcal{A}$  can obtain  $g^s$  after eavesdropping on the channel, by calculating  $e(g, g^s)$ , he can get  $e(g, g)^s$ , that is, the adversary  $\mathcal{A}$  knows  $e(g, g)^a$  and  $e(g, g)^s$ , wants to calculate  $e(g, g)^{as}$ . This is a **CDH** problem, there is no polynomial time algorithm to break it, so

$$Adv = |Pr[\mathcal{A} \text{ success}]| < \epsilon$$

#### 4. Experiments and Analysis

In this section, we mainly analyze the efficiency of our scheme and compared it with the Fully secure CP-ABE scheme proposed by Allison et al. [33] in setup, key generation, encryption, decryption, and memory consumption. The experiment is in the win10, 16 GB, AMD Ryzen 5 R2600 Six-Core 3.40 GHz platform. We choose to use the JPBC library of JAVA to build the environment and generate a composite order group with a size of 512 bits and an integer cyclic group with a size of 258 bits through an 83-bit elliptic curve. The data were obtained by running the experiments on a locally set up environment and were saved in a text file in “.xlsx” format. The figures were generated by comparing the data using MATLAB plotting.

Figure 5 shows the setup comparison between our scheme and Allison et al.’s scheme. Since the complexity of the setup is  $O(N)$  ( $N$  represents the number of attributes in the attribute universe), the time efficiency is almost the same except for computer errors.

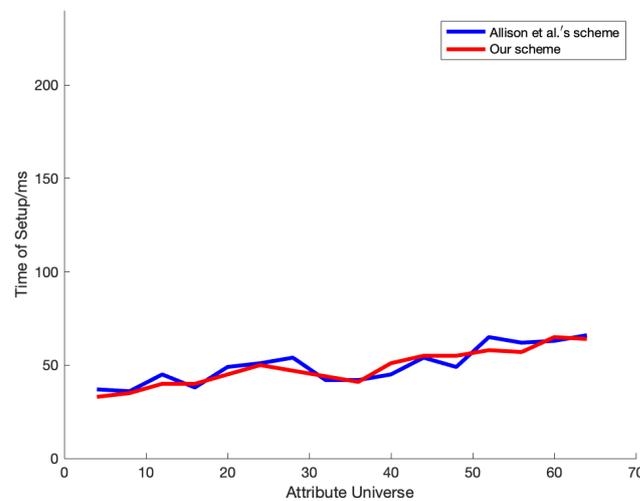


Figure 5. Setup.

The master key is in the form of a key–value pair:

```

MSK:
alpha: 210810353108659863024409106247517618452769941479846636980134442864523125
95818033429445987282464226795828802774079330

g3: 507123706182628610741111547764849270218939290666858116887669480037266931236
6558956079248951460266712797586740186721012, 3848333923503209101783513197106326
835379604308979589246948604032780415086659341351001892795149863403912326337017
537761, 0

beta: 3971351897302668818568847385425920497495147741445579066512932780617650627
246730432329467425793820791111050407589402
    
```

Figure 6 shows a comparison of the key generation time between our scheme and the scheme proposed by Allison et al. Our scheme involves interactive key generation, resulting in higher overhead compared to Allison’s scheme when the attribute space is small. However, as the attribute space grows, the performance gap between the two schemes decreases.

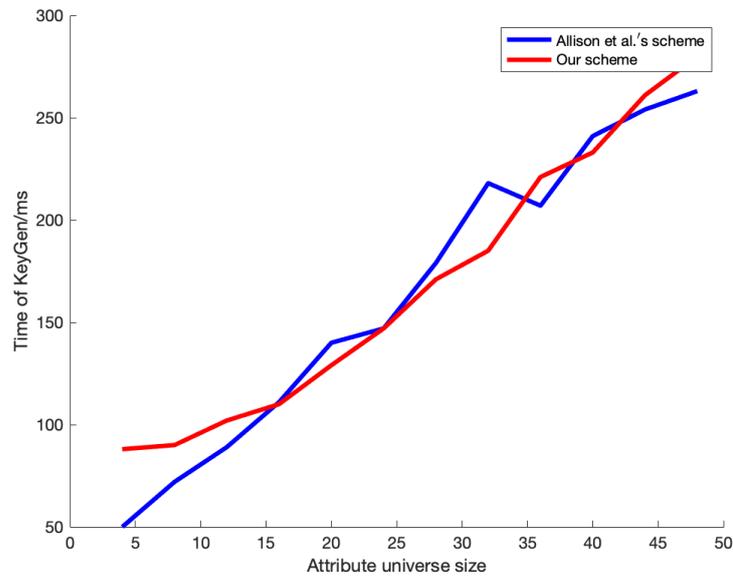


Figure 6. Key Generation.

Figure 7 illustrates a comparison of the encryption time between our scheme and the scheme proposed by Allison et al. The ciphertext complexity of Allison et al.'s scheme is  $O(C + N)$ , and the complexity of our scheme is also  $O(C + N)$ , where  $C$  represents the length of the ciphertext and  $N$  represents the number of attributes in the attribute space. In fact, Allison et al.'s scheme involves  $C + 2N$  terms, whereas ours involves  $C + 3N$  terms, which results in a small difference in overhead.

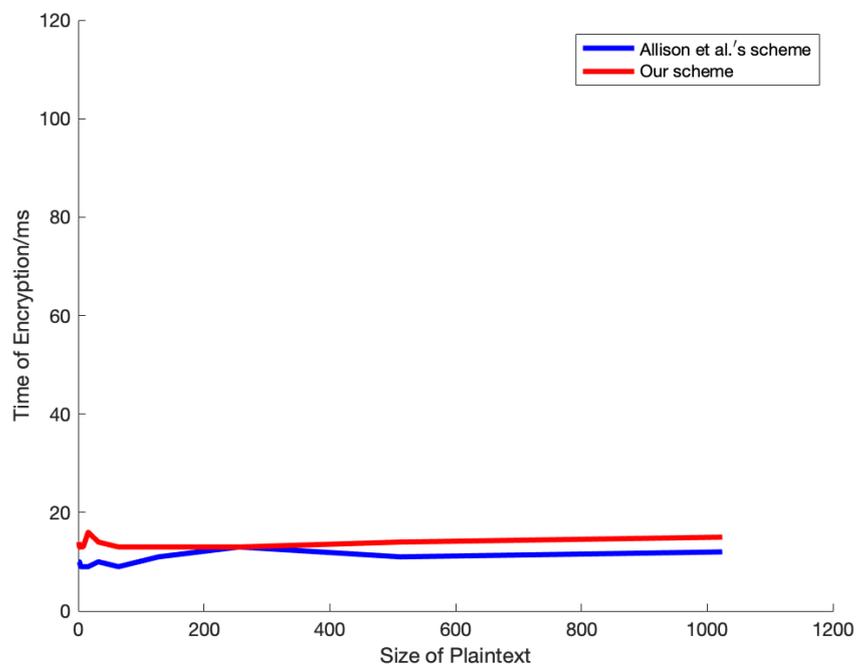


Figure 7. Encryption.

The generated ciphertext is also stored in the form of key–value pairs. We take plaintext "hello" as an example, and the encrypted  $C_0$  format is as follows:

```

CT_AT:
C0_0: {x=18512619911661450195327750867443546794232624419671207238173221146535506
95552872207298169171836981828215064641158984234, y=70592147653957222146120442588
2624722810944092606551932879619145643069907176528665887436471753185725971617607
9123508797}

C0_1: {x=10305743198129175737055428555819461127421625147813759967731163938926816
650740214215836427503578249751830703585598710737, y=9864469019751328170985532932
3260154136345304295214862269824474212174107344885351182673330718902818130451275
75927761095}

C0_2: {x=85627632985010802758634299337508008509965737271381026168608890332725357
51769387446114503063186080075196481905182611275, y=69157043429120428487195645886
4787393452354723338419697486692137357841812076088296305720924512144895097136125
4706491542}

C0_3: {x=85627632985010802758634299337508008509965737271381026168608890332725357
51769387446114503063186080075196481905182611275, y=69157043429120428487195645886
4787393452354723338419697486692137357841812076088296305720924512144895097136125
4706491542}

C0_4: {x=10828209153804955834077646467569440299821102129146337294704720899926979
6582045437576244731444812151580842960742884772, y=411045008855643689234607591514
8105748998457729928230076680665408791706458037634503664240890763024397642409757
902239244}
    
```

Figure 8 presents a comparison of the decryption time between our scheme and the scheme proposed by Allison et al. The time overhead is mainly focused on computing the secret  $s$ , so apart from computational errors, there is no difference in overhead.

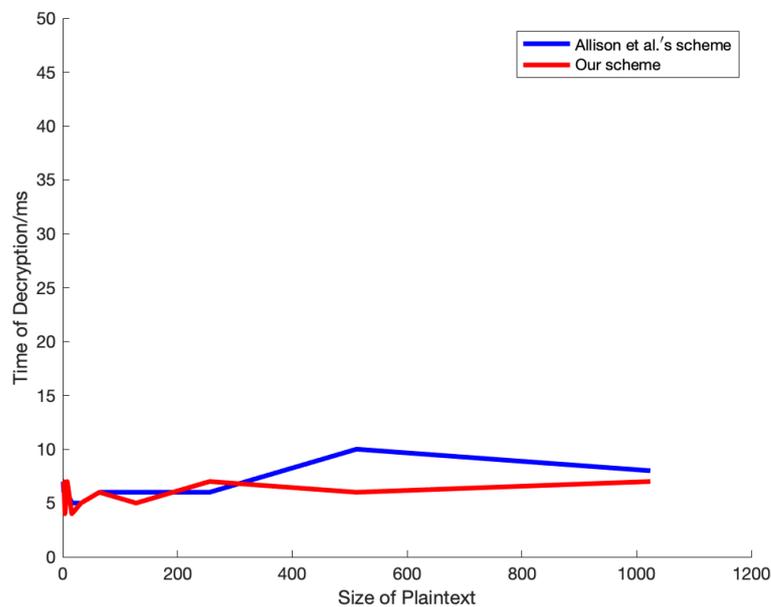


Figure 8. Decryption.

The plaintext obtained by decrypting the above ciphertext is as follows:

```
ourScheme.Decrypt("file/Key", "file/CT_AT");
The plaintext after decryption is:hello
```

Finally, Figure 9 shows a comparison of the memory overhead between our scheme and the scheme proposed by Allison et al. Due to the involvement of our scheme's interactive functions, such as *sendToAT()*, *KeyGenAT()*, and *id* extraction function *extractID()*, our scheme incurs a higher memory overhead than Allison et al.'s scheme.



**Figure 9.** Memory overhead comparison.

## 5. Conclusions

Based on Paillier encryption and *CP-ABE*, this paper proposes a data rights confirmation scheme, which effectively solves the ownership of data and the right to use data. The work of confirming the rights of the plaintext is transferred to the confirmation of the ciphertext. Before the plaintext audit, no one can access the original data except the user designated by the data user, which provides a guarantee that the source data will not be leaked. Finally, the security of the scheme is proved, and the efficiency and feasibility of the scheme are analyzed through experiments. First, this article does not exclude the third party, and the data transaction behavior between users is not specified, but only stipulates the "ownership" of data and the "rights to use" of data. Second, this paper does not consider how to implement the tracking of the key and revocation of the key when the data owner discovers the data leakage. Finally, there is no guarantee that the credentials stored in *AT* are authentic. In future work, we will conduct related research on data transactions. Considering the key tracing and key revocation problems, the traceability and revocability of keys can be achieved by combining the scheme proposed by Ning et al. [14] and the subset coverage technique. How to combine smart contracts and blockchain to solve the problem of traditional third-party untrustworthiness is also part of our future work.

**Author Contributions:** L.Z. designed the article architecture, wrote the manuscript, and completed the experimental tests. Y.C. provided funding support, Z.H., Y.L. and T.L. checked the manuscript and the experimental data. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by Foundation of National Natural Science Foundation of China (61962009 and 62202118), Natural Science Research Technology Top Talent Project of Education Department of Guizhou Province ([2022]073), and the vocational education science research project of Education Department of Guizhou Province.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

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

## References

1. Sheng, H.; Cong, R.; Yang, D.; Chen, R.; Wang, S.; Cui, Z. UrbanLF: A Comprehensive Light Field Dataset for Semantic Segmentation of Urban Scenes. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 7880–7893. [\[CrossRef\]](#)
2. Chen, Y.; Sun, J.; Yang, Y.; Li, T.; Niu, X.; Zhou, H. PSSPR: A Source Location Privacy Protection Scheme Based on Sector Phantom Routing in WSNs. *Int. J. Intell. Syst.* **2022**, *37*, 1204–1221.
3. Lv, Z.; Song, H. Mobile internet of things under data physical fusion technology. *IEEE Internet Things J.* **2019**, *7*, 4616–4624.
4. Meng, F.; Xiao, X.; Wang, J. Rating the crisis of online public opinion using a multi-level index system. *arXiv* **2022**, arXiv:2207.14740.
5. Cao, B.; Zhao, J.; Lv, Z.; Yang, P. Diversified Personalized Recommendation Optimization Based on Mobile Data. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 2133–2139. [\[CrossRef\]](#)
6. Chen, T.; Yu, Y.; Duan, Z.T. Blockchain/ABE-based Fusion Solution for E-government Data Sharing and Privacy protection. In Proceedings of the EITCE 2020: 2020 4th International Conference on Electronic Information Technology and Computer Engineering, Xiamen, China, 6–8 November 2020.
7. Li, T.; Wang, Z.; Chen, Y.; Li, C.; Jia, Y.; Yang, Y. Is semi-selfish mining available without being detected? *Int. J. Intell. Syst.* **2021**, *37*, 10576–10597.
8. Heidari, A.; Navimipour, N.J.; Unal, M. A Secure Intrusion Detection Platform Using Blockchain and Radial Basis Function Neural Networks for Internet of Drones. *IEEE Internet Things J.* **2023**, early access. [\[CrossRef\]](#)
9. Waters, B.R.; Sahai, A. Fuzzy identity based encryption. In Proceedings of the Advances in Cryptology—EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005. [\[CrossRef\]](#)
10. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for finegrained 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. [\[CrossRef\]](#)
11. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 20–23 May 2007. [\[CrossRef\]](#)
12. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Proceedings of the International Workshop on Public Key Cryptography, Taormina, Italy, 6–9 March 2011. [\[CrossRef\]](#)
13. Lewko, A.; Waters, B. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *Annual Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2012. [\[CrossRef\]](#)
14. Ning, J.; Dong, X.; Cao, Z.; Wei, L. *Accountable Authority Ciphertext-Policy Attribute-Based Encryption With White-Box Traceability and Public Auditing in the Cloud*; Springer: Cham, Switzerland, 2015. [\[CrossRef\]](#)
15. Cao, B.; Sun, Z.; Zhang, J.; Gu, Y. Resource allocation in 5G IoV architecture based on SDN and fog-cloud computing. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 3832–3840.
16. Yu, G.; Wang, Y.; Cao, Z.; Lin, J.; Wang, X. Traceable and undeniable ciphertext-policy attribute-based encryption for cloud storage service. *Int. J. Distrib. Sens. Netw.* **2019**, *15*. [\[CrossRef\]](#)
17. Yuan, F.; Chen, S.; Xu, K.L.L. *Research on the Coordination Mechanism of Traditional Chinese Medicine Medical Record Data Standardization and Characteristic Protection under Big Data Environment*; Shandong People’s Publishing House: Jinan, China, 2021.
18. Chen, B.; Hu, J.; Zhao, Y.; Ghosh, B.K. Finite-Time Velocity-Free Rendezvous Control of Multiple AUV Systems With Intermittent Communication. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 6618–6629. [\[CrossRef\]](#)
19. Lu, S.; Ban, Y.; Zhang, X.; Yang, B.; Yin, L.; Liu, S.; Zheng, W. Adaptive control of time delay teleoperation system with uncertain dynamics. *Front. Neurobot.* **2022**, *152*, 928863.
20. Peng, Y. Research on authenticating data rights in Big Data environment. *Mod. Sci. Technol. Telecommun.* **2016**, *46*, 17–20. [\[CrossRef\]](#)
21. Guo, B.; Li, Q.; Duan, X.L.; Shen, Y.C.; Dong, X.Q.; Zhang, H.; Shen, Y.; Zhang, Z.L.; Luo, J. Personal Data Bank: A New Mode of Personal Big Data Asset Management and Value-Added Services Based on Bank Architecture. *Chin. Comput.* **2017**, *40*, 126–143.
22. Wang, S.; Li, C. A Big Data Right Confirmation Method and System Based on Blockchain Technology. Patent CN106815728A, 9 June 2017.
23. Wang, H.; Tian, Y.; Yi, X. Blockchain-based Big Data Right Confirmation Scheme. *Comput. Sci.* **2018**, *45*, 6.
24. Zhao, H.; Zhao, B.; Cheng, S. The Mechanism of Confirming Big Data Property Rights Based on Smart Contract. In Proceedings of the 2019 4th International Conference, Jinan, China, 18–21 October 2019. [\[CrossRef\]](#)
25. Zhou, G.; Yan, B.; Wang, G.; Yu, J. Blockchain-Based Data Ownership Confirmation Scheme in Industrial Internet of Things. In Proceedings of the Wireless Algorithms, Systems, and Applications: 16th International Conference, WASA 2021, Nanjing, China, 25–27 June 2021; Part I; Springer International Publishing: Cham, Switzerland, 2021; pp. 121–132.
26. Dai, X.; Xiao, Z.; Jiang, H.; Alazab, M.; Lui, J.C.; Min, G.; Dustdar, S.; Liu, J.; Task Offloading for Cloud-Assisted Fog Computing With Dynamic Service Caching in Enterprise Management Systems. *IEEE Trans. Ind. Inform.* **2023**, *19*, 662–672. [\[CrossRef\]](#)
27. Liu, Y.; Zhang, Y.; Yang, Y.; Ma, Y. DOCS: A Data Ownership Confirmation Scheme for Distributed Data Trading. *Systems* **2022**, *10*, 226.
28. Damgrd, I.; Thorbek, R. Linear integer secret sharing and distributed exponentiation. In Proceedings of the Public Key Cryptography—PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, 24–26 April 2006. [\[CrossRef\]](#)

29. Liu, Z.; Cao, Z.; Wong, D.S. Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on ebay. In Proceedings of the 2013 ACM Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013. [[CrossRef](#)]
30. Boneh, D.; Goh, E.; Nissim, K. *Evaluating 2-DNF Formulas on Ciphertexts*; Springer: Berlin/Heidelberg, Germany, 2005. [[CrossRef](#)]
31. Lewko, A.B.; Waters, B. Decentralizing Attribute Based Encryption. In Proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, 15–19 May 2011. [[CrossRef](#)]
32. Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the EUROCRYPT'99, Prague, Czech Republic, 2–6 May 1999. [[CrossRef](#)]
33. Allison Lewko, B.; Okamoto, T.; Sahai, A.; Takashima, K.; Waters, B. Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In Proceedings of the International Conference on Theory and Applications of Cryptographic Techniques, French Riviera, France, 30 May–3 June 2010; Springer: Berlin/Heidelberg, Germany, 2010. [[CrossRef](#)]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.