Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6662662, 11 pages
https://doi.org/10.1155/2020/6662662

Research Article

Balancing Access Control and Privacy for Data Deduplication via Functional Encryption

Bo Mi,[1] Ping Long ,[1] Yang Liu,[1] and Fengtian Kuang[2]

[1]*College of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China*
[2]*College of Mathematics and Statistics, Chongqing Jiaotong University, Chongqing 400074, China*

Correspondence should be addressed to Ping Long; longpingcq@163.com

Received 2 November 2020; Revised 19 November 2020; Accepted 24 November 2020; Published 10 December 2020

Academic Editor: Yong Chen

Data deduplication serves as an effective way to optimize the storage occupation and the bandwidth consumption over clouds. As for the security of deduplication mechanism, users' privacy and accessibility are of utmost concern since data are outsourced. However, the functionality of redundancy removal and the indistinguishability of deduplication labels are naturally incompatible, which bring about a lot of threats on data security. Besides, the access control of sharing copies may lead to infringement on users' attributes and cumbersome query overheads. To balance the usability with the confidentiality of deduplication labels and securely realize an elaborate access structure, a novel data deduplication scheme is proposed in this paper. Briefly speaking, we drew support from learning with errors (LWE) to make sure that the deduplication labels are only differentiable during the duplication check process. Instead of authority matching, the proof of ownership (PoW) is then implemented under the paradigm of inner production. Since the deduplication label is light-weighted and the inner production is easy to carry out, our scheme is more efficient in terms of computation and storage. Security analysis also indicated that the deduplication labels are distinguishable only for duplication check, and the probability of falsifying a valid ownership is negligible.

1. Introduction

As a flourishing service mode, cloud computing adopts load balancing, distributed computing, and other technologies to conveniently provide computation and storage functions for remote follow-up users, thus saving local resources and promoting work efficiency. However, if the users immoderately outsource their data to the cloud, a serious problem may occur due to massive duplicated data. As reported in [1], almost half of the cloud storage is wasted because of data redundancy. Consequently, the budget for managing duplicate data raises up to eight times than that of source data maintenance [2, 3]. With the explosive growth of data nowadays, the tremendous storage requirements or the exorbitant administrative expenses have put enormous pressure on cloud service providers. Therefore, how to store and manage data economically and efficiently has become a serious challenge for these enterprises.

To cut down the costs caused by redundant data, deduplication technology has been widely used by cloud service providers [4]. In such a technology, duplication check and proof of ownership are two key problems. Till now, the problem of how to balance the conflict between comparability and confidentiality for secure duplication check remains unsolved [5]. Meanwhile, the problems of how to efficiently validate the access authority and how to achieve complex access structures are also urgent to address, considering that the mechanism of query matching is cumbersome and the downloading certificates may be abused to launch various attacks.

As a research hotspot, lots of attentions are put on the efficiency and security of data deduplication. In the published literature, Li et al. [6] suggested carrying out deduplication by comparing the fingerprint of the outsourced file with the uploaded ones in a direct way. However, this method is deficient since the communication and comparison of those fingerprints are inefficient and the contents

of data are exposed. To reduce the traffic of deduplication labels and conceal the data, Puzio et al. [7] used the hash function to code the same plaintexts into identical values, which serve as the labels for duplication check. Although this method achieved the goals of transmission efficiency and storage saving, it is vulnerable to dictionary attacks since the hash values are overt.

In order to ensure the confidentiality of deduplication labels, Chen et al. [8] utilized the message lock encryption (MLE) to encrypt those hash values of data. However, the traditional MLE scheme is not semantic secure and vulnerable against quantum attacks [9].

Fortunately, cryptographers have been devoted to design secure, efficient, and effective crypto systems to resist quantum attacks in recent years. In 2005, Regev et al. [10] proposed a novel paradigm as an underpinning of cryptography, namely, learning with errors. They proved that the difficulty of solving it is equivalent to the hardness of shortest vector problem (SVP) over lattice, and thus, it can resist the attacks based on quantum computing. Besides, it is provided with the capacity of homomorphic and linear computation. Therefore, we consider exploiting it in our scheme to ensure the functionality, efficiency, and security of deduplication labels.

As for the proof of ownership, the best solutions till now are all based on Merkle hash tree (MHT) [11, 12]. In detail, the cloud and the user independently hold an MHT computed from the outsourced data. Thus, the user can upload the same MHT to the cloud for comparison. The disadvantages of such scheme are not only high storage and communication overheads but also low computation efficiency. Therefore, Chen et al. [13] improved it by randomly asking the cloud to select some leaf nodes of the MHT to challenge the user. The user must trace the path from the root to these leaves as a reply to prove that he possesses the same tree. Although this method does not require the transmission of the whole MHT for comparison, it demanded that the user and the cloud should construct and store a complete MHT for each file. Moreover, the challenge-response mode implies a long delay.

In order to promote the performance of PoW, the advantages of inner product predicate gradually entered the researchers' sight [14–16]. Roughly speaking, only if the inner product results 0, the user can be granted a permission to access the corresponding file. The most significant merit of this method is using computation instead of comparison to efficiently perform ownership proof. Therefore, we adopted it in our scheme to balance the conflict between the variety of access structures and the security of users' privacy.

Aiming at checking replication over semantic secure deduplication labels and achieving fine-grained access control, this paper proposed a novel cloud data deduplication scheme by exploiting LWE (learning with errors) together with inner product predicate. Our contributions are abbreviated as follows:

(i) Though designed for the purpose of deduplication, the deduplication labels are indistinguishable to any process except for duplication check. This property is achieved in virtue of semantic secure and homomorphic LWE, which is also resistant to quantum attack.

(ii) The proof of ownership is carried out by inner product, which is computationally efficient. Besides, we impose the accessibility of users on their attributes, implying the functionality of the elaborate access structure and ownership transfer.

(iii) For each file, only one light-weighted downloading certificate should be stored by the cloud, while the clients should only carry out and upload its corresponding proof on demand. That is to say that both the storage and bandwidth are economic for cross-user access.

The rest of this paper is organized as follows. In Section 2, some formal definitions related to LWE and inner product predicate are given. Section 3 depicts our deduplication scheme, including the detailed way for duplication check and ownership proof. The correctness of our scheme is formally validated in Section 4, followed by security and performance analysis in Sections 5–7 that concludes the paper.

2. Preliminaries

For better understanding of our scheme, the concepts related to learning with errors and inner product predicate [2, 17] will be introduced in advance.

Definition 1 (Integer lattice). An integer lattice Λ is the integer linear combination of vectors $a_1, a_2, \ldots, a_k a_1, a_2, \ldots, a_k$ over $\mathbb{Z}^m$, expressed as

$$\Lambda(a_1, a_2, \ldots, a_k) = \left\{ \sum_{i=1}^{k} a_i z_i : z_i \in \mathbb{Z} \right\}. \tag{1}$$

Definition 2 (LWE hardness assumption). On parameters n, m, q, α and a discrete Gaussian distribution χ, where

$$\Pr[x \longleftarrow \chi : |x| > \alpha q] < \mathrm{negl}(n), \tag{2}$$

for $x \in \mathbb{Z}_q$, we select a noise e from χ^m and uniformly sample a vector $s \in \mathbb{Z}_q^n$ together with a matrix $P \in \mathbb{Z}_q^{n \times m}$. Based on the value of

$$b = [sP + e]_q, \tag{3}$$

two versions of LWE hardness can be defined as follows:

(a) LWE-Search hardness: Given multiple pairs of (P, b) on constant P and s, searching for the value of s is difficult.

(b) LWE-Determination hardness: For uniformly sampled $b' \in \mathbb{Z}_q^n$, the tuples of (P, b) and (P, b') are statistically indistinguishable. It means that it is difficult to tell if the second term of those tuples are randomly chosen or computed from formula (3).

In fact, the LWE-search hardness is equivalent to the problem of finding a short enough vector in lattice (GapSVP), and the LWE-determination hardness can be reduced to the problem of solving linearly independent shortest vectors (SIVP) of a lattice in the worst case. Therefore, the LWE assumption can be used to guarantee the one-way property for encryption with semantic security.

Definition 3 (Inner product predicate). The inner product predicate $P_{n,q}$ is defined on the Cartesian product $K \times I$ that

$$P_{n,q}(\overrightarrow{v}, t\overrightarrow{w}) = \begin{cases} 1, & \sum_{i=1,\ldots,n} v_i w_i = 0, \\ 0, & \text{other.} \end{cases} \tag{4}$$

From the perspective of functional encryption (FE), I can be deemed as the space of ciphertexts and K is composed of secret keys. Once a correct key $\overrightarrow{v}$ is known, we are able to learn the output of function $P_{n,q}(\overrightarrow{v}, t\overrightarrow{w})$.

To construct an attribute-based access control policy, the access structure is coded as a vector $\overrightarrow{w}$, thus the access authority can be verified with respect to the consistency of authorization certificate $\overrightarrow{v}$.To avoid obfuscation, the symbols used in this paper is listed in advance, as in Table 1.

3. Duplication Check Based on LWE

To prevent dictionary attacks caused by the exposure of deduplication labels, we intended to make them indistinguishable except for the process of duplication check. Therefore, LWE is adopted to randomize the hash value of file to ensure the indistinguishability of deduplication labels and resist the attacks of quantum computation. In addition, we exploit inner product predicate to control the accessibility of clients, which is flexible for functions such as cross-user sharing and ownership transfer. The logical idea of our scheme is illustrated below, which is shown in Figure 1.

4. File Upload

A user denoted as A, who possesses a file M_A and expects to upload it, is not aware of its existence over cloud at the very beginning. To avoid unnecessary storage and bandwidth, he is supposed to check if there is a copy already held by the server.

Drawing support from any strong-collision resistant hash function

$$H\colon \{0,1\}^* \longrightarrow \{0,1\}^{\ell}, \tag{5}$$

the user figures out the hash value of file M_A as

$$h_A = H(M_A), \tag{6}$$

and codes it as a vector of ℓ elements. On fixed public matrix $P \in \mathbb{Z}_q^{n\times m}$ and a pseudorandom sequence generator (PSRG), he produces a vector

$$\overrightarrow{s}_A = (\mathrm{PSRG}(h_A), -1) \in \mathbb{Z}_q^{n+1}, \tag{7}$$

and exploits LWE to obtain

$$\overrightarrow{v_A} = (P, b_A) \cdot r_A \in \mathbb{Z}_q^{n+1}. \tag{8}$$

Herein, $b_A = [\mathrm{PSRG}(h_A)P + e_A]_q$ stands for the last row of (P, b_A), where $\mathrm{PSRG}(h_A)$ is considered as a n dimensional vector and r_A is randomly chosen from $\{-1, 0, 1\}^m$.To this point, the user is ready to take the $n+1$ dimensional vector $\overrightarrow{v_A}$ as a deduplication label and upload it to the cloud. Since the subsequent actions he should take depend directly on the result of duplication check, we will discuss the situations for original uploader and repeated uploader, respectively, who are denoted as A and B for clarity.

4.1. The Process of Original Uploading. We defer the description of duplication check to the circumstance of repeated upload, if suppose that user A is informed with the inexistence of file M_A. For further deduplication, he should secretly upload the deduplication certificate $\overrightarrow{s_A}$ to the cloud. To ensure the confidentiality of his file, its hash value can be taken as a symmetric key $\mathrm{sk}_A = h_A$ to hide the plaintext as

$$\mathrm{Enc}_{\mathrm{sk}_A}(M_A) = C_A. \tag{9}$$

Then, the cloud preserves the uploaded ciphertext C_A for storage and the deduplication certificate $\overrightarrow{s_A}$ for duplication check. To further retrieve the file, user A ought to upload a downloading certificate as well, like the following.

Assuming that the attributes of user A correspond to a secret vector $\overrightarrow{\mu_A} = (\mu_{A,0}, \mu_{A,1}, \ldots, \mu_{A,n-1}) \in Z_q^n$, which can also be regarded as a polynomial

$$f(\overrightarrow{\mu_A}) = \mu_{A,0} + \mu_{A,1}x + \cdots + \mu_{A,n-1}x^{n-1} \bmod q. \tag{10}$$

It is worth mentioning that the user is aware of the elements of $\overrightarrow{\mu_A}$ only if he corresponds to those attributes. To actualize a functional encryption which reflects the access structure in covert manner, he uniformly samples two vectors $w_i\,(i = 0, 1, \ldots, n-2)$ and $\overrightarrow{u_A} = (u_{A,0}, u_{A,n-1}, u_{A,n-2}, \ldots, u_{A,1})$. Similarly, the vector $\overrightarrow{u_A} = (u_{A,0}, u_{A,n-1}, \ldots, u_{A,1}) \in Z_q^n$ can also be expressed as a polynomial $g(\overrightarrow{u_A}) = u_{A,0} + u_{A,n-1}x + \cdots + u_{A,1}x^{n-1} \bmod q$, which is equivalent to a cyclic matrix

$$U_A = \begin{bmatrix} u_{A,0} & u_{A,n-1} & \cdots & u_{A,1} \\ u_{A,1} & u_{A,0} & \cdots & u_{A,2} \\ \vdots & \vdots & \cdots & \vdots \\ u_{A,n-1} & u_{A,n-2} & \cdots & u_{A,0} \end{bmatrix} \in Z_q^{n\times n}, \tag{11}$$

with respect to the homorganic between polynomials and cyclic matrices.

In order to construct the correct downloading certificate, he computes $U_A \cdot \overrightarrow{\mu_A} = \overrightarrow{X_A} = (x_{A,0}, x_{A,1}, \ldots x_{A,n-1})^T$ and figures out w_{n-1} for $<\overrightarrow{X_A}, \overrightarrow{w}> = 0 \bmod q$ by

$$w_{n-1} = -\frac{\sum_{i=0}^{n-2} x_{A,i} w_i}{x_{n-1}} \bmod q. \tag{12}$$

Table 1: Symbols and notations.

Symbol	Notation
f	Length of a file
b	Length of each file block
g	Length of the hash value
N	Number of users participating in key aggregation
K	Total number of bloom filters
L	Length of bloom filter array
$Q(n)$	Number of common attributes in a user group
n	Number of attributes for an individual user
$Hash$	Computational cost of performing a hash function
CE_K	Computational cost of performing a key aggregation
Enc	Computational overhead of performing a symmetric encryption
PoW	Computational cost of performing a proof of ownership
Add	Computational cost of performing an addition

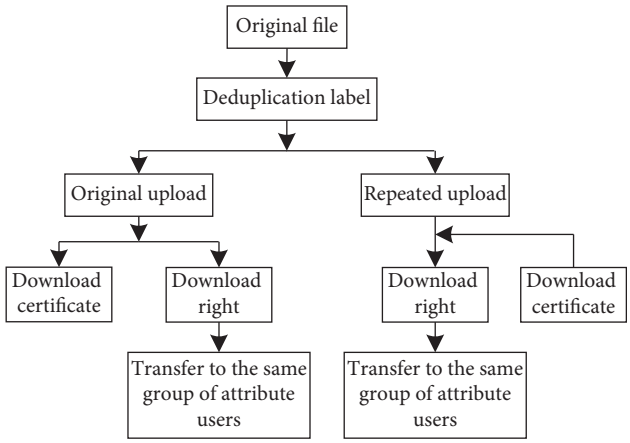

Figure 1: The overall framework of the deduplication program.

After that, the user uploads $\overrightarrow{w} = (w_0, w_1, \ldots, w_{n-1})$ as the downloading certificate and submits

$$y = w_{n-2}x_{A,n-2} + w_{n-1}x_{A,n-1} = -\left(\sum_{i=0}^{n-3} w_i x_{A,i}\right), \tag{13}$$

to the cloud for further expansions on access structure.

At the end, the user preserves the hash value sk_A, the essential elements $\overrightarrow{u_A} = (u_{A,n-1}, \ldots, u_{A,1}, u_{A,0})$ of matrix U_A, and the replied link of outsourced file. While the ciphertext C_A can be held by the cloud server, attached with $\overrightarrow{s_A}$, y, and $\overrightarrow{w}$ for duplication check, access expansion, and ownership proof.

4.2. The Process of Repeated Uploading. As mentioned before, once a deduplication label is figured out, any user should firstly hand it over to the cloud for duplication check. Assume that the deduplication certificate of an existing file M_A is $\overrightarrow{s_A}$, the cloud can inspect its consistency with another deduplication label $\overrightarrow{v_B}$ as the following:

When user B expects to upload his file M_B, he submits its deduplication label $\overrightarrow{v_B} = (P, b_B) \cdot r_B$ to the cloud and keeps the hash value h_B private.

Based on an outsourced deduplication certificate $\overrightarrow{s_A}$, the cloud computes within a lifted interval $[-(q-1)/2, (q-1)/2]$ which is as follows:

$$< \overrightarrow{s_A}, \overrightarrow{v_B} > \; = \; < \mathrm{PSRG}(h_A)P - \mathrm{PSRG}(h_B)P - e_B, r_B > . \tag{14}$$

It can be seen that, if the two files are identical, only $\langle -e_B, r_B \rangle$ will remain in formula (14). Therefore, when the result satisfies

$$\left\| \langle \overrightarrow{s_A}, t\overrightarrow{v_B} \rangle \right\|_\infty \leq \alpha q, \tag{15}$$

the cloud can ensure the duplication of file M_B with negligible false positive.

To validate his accessibility, user B should also figure out the downloading right of the corresponding file. However, it is more reasonable to use existing download rights $\overrightarrow{w}$ held by the cloud server for the purpose of storage saving. Based on this, user B can use the following subprotocol to obtain the download right of the duplicate file, and the cloud will simply send the link back to him for further retrieval.

4.3. The Subprotocol for Access Expansion. Denoting the secret corresponding to the attributes of repeated uploader B as $\overrightarrow{\mu_B} = (\mu_{B,0}, \mu_{B,1}, \ldots, \mu_{B,n-1})^T$. To bind the access structure with his own attributes, he should also figure out a cyclic matrix U_B which can be used to compute his proof of ownership which is as follows:

$$< U_B \cdot \overrightarrow{\mu_B}, \overrightarrow{w} > \; = 0. \tag{16}$$

Though the downloading certificate $\overrightarrow{w}$ cannot be exposed to prevent unauthorized access, the cloud can provide user B with the values of $(w_{n-3}, w_{n-2}, w_{n-1})$ and y to help him calculate the correct cyclic matrix U_B. Thus, the downloading right can be carried out by user B in Algorithm 1.

5. Proof of Ownership

Once any legal user obtained his downloading right, he should be authorized to retrieve the corresponding file from the cloud. To improve the efficiency of ownership proof, access authorization is executed in a computational way.

After uploading, the legal user A will be provided with the last row $\overrightarrow{u_A} = (u_{A,n-1}, \ldots, u_{A,1}, u_{A,0})$ of the cyclic matrix. Therefore, he only needs to form the cyclic matrix U_A and combines it with his attribute vector $\overrightarrow{u_A}$ to figure out the downloading right. Based on the resulted vector, the cloud can easily verify his accessibility by functional encryption. The process of PoW is completely given in Algorithm 2.

After obtaining the ciphertext C_A, user A can decrypt the file by computing $\mathrm{Dec}_{\mathrm{sk}_A}(C_A) = M_A$ because he is aware of the secret key $\mathrm{sk}_A = H(M_A)$.

In fact, the ownership proof process for user B is similar to that of user A. The reason why user B can also decrypt the file C_A is due to the equality of plaintexts M_A and M_B. Since

$\mathrm{sk}_B = H(M_A)$ he is able to obtain the corresponding file via $\mathrm{Dec}_{\mathrm{sk}_B}(C_A) = M_A$.

6. Downloading Right Transfer

On noting that, without the secret vectors corresponding to the attributes of legal users, other users are incapable of computing the downloading right even if the last row of cyclic matrix is known. Since the access controls subprotocol, any legal user can directly transfer the resultant downloading right to other users to avoid redundant operations such as peer to peer transmission. However, it may lead to the abuse of downloading right and violate the confidentiality of user's attributes. Practically, legal users are prone to transfer the downloading right of their file to others who share party of common attributes with him. Therefore, we designed a protocol that any legal user can update the downloading right and transfer it to a group of users with the same set of attributes. In this way, the owner does not have to download the file from the cloud and only needs to transfer the downloading right to other users to complete file sharing, which effectively reduces the consumption of communication bandwidth.

Definition 4 (Common attributes vector). Suppose that the file owner A can be identified by attributes vector $\overrightarrow{\mu_A} = (\mu_{A,0}, \mu_{A,1}, \ldots, \mu_{A,n-1})$, and all users in the same group have $Q(n)$ common attributes denoted by $\overrightarrow{\mu_{\mathrm{all}}} = (\mu_{\mathrm{all},0}, \ldots, \mu_{\mathrm{all},n-1}, \ldots, \mu_{\mathrm{all},Q(n)})$. Then, the common attributes vector $\overrightarrow{\mu_{\mathrm{team}}} = (\mu_{\mathrm{team},0}, \ldots, \mu_{\mathrm{team},n-1})$ can be defined as a partial ordering relation that $\mu_{\mathrm{team},i} = \mu_{A,i}$ if $\mu_{A,i} \in \{\mu_{\mathrm{tall},j} | j = 0, \ldots, Q(n)\}$ and $\mu_{\mathrm{team},i} = 0$, otherwise.

Specifically, the process that the user A constructs the common attribute vector $\overrightarrow{\mu_{\mathrm{team}}} = (\mu_{\mathrm{team},0}, \mu_{\mathrm{team},1}, \ldots, \mu_{\mathrm{team},n-1})$ is detailed in Figures 2 and 3.

As shown in Figures 2 and 3, the user A mainly retains the secret attributes shared by the same group and sets the attributes which are distinct in the user group as 0. Finally, he outputs a common attribute vector $\overrightarrow{\mu_{\mathrm{team}}}$.

6.1. Proof of Ownership. The user A performs the following steps to realize the PoW and retrieves $(w_{n-3}, w_{n-2}, w_{n-1})$ and y.If the downloading right is valid, the inner product will result in 0, meaning that the user A is authorized to retrieve the file. Therefore, the cloud server returns $C_A(w_{n-3}, w_{n-2}, w_{n-1})$ and y back to him. Similarly, the values of $(w_{n-3}, w_{n-2}, w_{n-1})$ and y can be used to update the downloading right for a group of users. Specifically, the process of PoW is shown in Algorithm 2, which is the same for any valid user even if the updated downloading right is used.

6.2. Update the Downloading Right. To share the file to a group, the downloading right update process can be carried out by the user A as the following. In a clear form, the process that the user A calculates the downloading right for a group of users is shown in Algorithm 3.

6.3. Sharing the Downloading Right. After the previous two stages, the user A can share the vector $\overrightarrow{u'_{\mathrm{team}}} = (u'_{\mathrm{team},n-1}, \ldots, u'_{\mathrm{team},1}, u'_{\mathrm{team},0})$ and the secret key sk_A to all users who are within the same attributes set. In these ways, a group of users are provided with the downloading right, which can be valid if the common attributes vector $\overrightarrow{\mu_{\mathrm{team}}}$ is known.

7. Correctness Proof

The previous section is mainly composed of three parts, namely, the file uploading, the proof of ownership, and the downloading right transfer. To verify the correctness of our design, this section intends to prove that file duplication can be effectively eliminated and only authenticated users can access the file.

Firstly, the correctness for the deduplication label is given by Theorem 1.

Theorem 1 (Correctness of deduplication label). *Suppose that the cloud holds a deduplication certificate $\overrightarrow{s_A}$ which is correspondent to file M_A. After the user B uploaded the deduplication label $\overrightarrow{v_B}$ before outsourcing the same file M_A, the cloud can perform deduplication correctly with negligible false positive.*

Proof. Due to the deduplication certificate $\overrightarrow{s_A} = (\mathrm{PSRG}(h_A), -1) \in \mathbb{Z}_q^{n+1}$ stored on the cloud, where $h_A = H(M_A)$. After the user B uploaded the deduplication label $\overrightarrow{v_B} = (P, b_B) \cdot r_B \in \mathbb{Z}_q^{n+1}$ of the same file M_A to the cloud for $b_B = [\mathrm{PSRG}(h_B) + e_B]$, the cloud executes the following calculation on each deduplication certificate. Once $\overrightarrow{s_A}$ is met, the inner product can be carried out as follows:

$$
\begin{aligned}
<\overrightarrow{s_A}, \overrightarrow{v_B}> &= \left(\left(\underbrace{\mathrm{PSRG}(h_A)}_{n\,\mathrm{bits}}, -1 \right) \cdot \left(\begin{bmatrix} \underbrace{P}_{n\times m} \\ \underbrace{b_B}_{1\times m} \end{bmatrix}^{(n+1)\times m} \cdot r_B \right) \right) \\
&= \left(\left(\underbrace{\mathrm{PSRG}(h_A)}_{1\times n}, -1 \right) \cdot \begin{bmatrix} \underbrace{P}_{n\times m} \\ \underbrace{b_B}_{1\times m} \end{bmatrix}^{(n+1)\times m} \right) \cdot r_B \\
&= (\mathrm{PSRG}(h_A) \cdot P - b_B) \cdot r_B \\
&= <\mathrm{PSRG}(h_A) \cdot P - (\mathrm{PSRG}(h_B) \cdot P + e_B), r_B> \\
&= <\mathrm{PSRG}(h_A) \cdot P - \mathrm{PSRG}(h_B) \cdot P - e_B, r_B>.
\end{aligned}
\tag{17}
$$

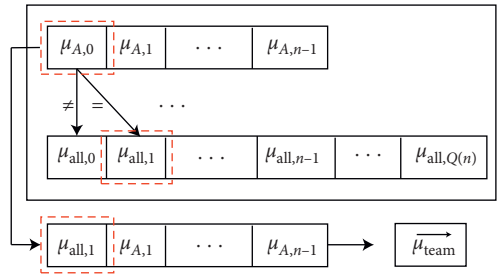

Figure 2: Common attributes.

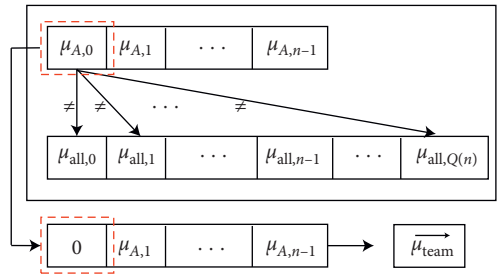

Figure 3: Noncommon attributes.

Since $\text{PSRG}(\cdot)$ is a deterministic algorithm, when $h_A = h_B$,$\text{PSRG}(h_A) = \text{PSRG}(h_B)$. Meanwhile, according to the common matrix P, it is obvious that $\text{PSRG}(h_A) \cdot P = \text{PSRG}(h_B) \cdot P$. Thus, we can easily see that $\langle \overrightarrow{s_A}, \overrightarrow{v_B} \rangle = \langle -e_B, r_B \rangle$ from equation (17). Because the inner product of $< -e_B, r_B > \leq (1/Q(m))$ is definite, the inner product of $< \overrightarrow{s_A}, \overrightarrow{v_B} > \leq (1/Q(m))$ can also be guaranteed, meaning that duplication can be detected with 100% probability.

Theorem 2 (Correctness of download right). *Suppose that the cloud possesses a downloading certificate $\overrightarrow{w}$ corresponding to file M_A, then any legal user can correctly pass the procedure of PoW in terms of his downloading right.*

Proof. For user A, who uploads the original file M_A to the cloud, he rotates $\overrightarrow{u_A} = (u_{A,n-1}, \ldots, u_{A,1}, u_{A,0})$ to right by one bit to get $\overrightarrow{u_A'} = (u_{A,0}, u_{A,n-1}, \ldots, u_{A,1})$ and uses it to reconstruct the cyclic matrix U_A. Then, user A calculates download right

$$\overrightarrow{X_A} = U_A \cdot \overrightarrow{\mu_A} = \left(x_{A,0}, x_{A,1}, \ldots x_{A,n-1}\right)^T, \tag{18}$$

where $\overrightarrow{\mu_A} = (\mu_{A,0}, \mu_{A,1}, \ldots, \mu_{A,n-1})$ are the attributes of the user A. After which the user A sends the download right $\overrightarrow{X_A}$ to the cloud. Finally, the cloud calculates the inner product of

$$< \overrightarrow{w}, \overrightarrow{X_A} > = \sum_{i=0}^{n-1} w_i \cdot x_{A,i} = \sum_{i=0}^{n-2} w_i \cdot x_{A,i} + w_{n-1} \cdot x_{A,n-1} = \sum_{i=0}^{n-2} w_i \cdot x_{A,i} + \left(-\sum_{i=0}^{n-2} w_i \cdot x_{A,i} \right) = 0. \tag{19}$$

Based on the last element of download certificate $\overrightarrow{w}$ is $w_{n-1} = -((\sum_{i=0}^{n-2} x_{A,i} w_i)/(x_{n-1})) \bmod q$, so that the result of $w_{n-1} x_{A,n-1}$ can transfer as $(-\sum_{i=0}^{n-2} w_i x_{A,i})$. Therefore, the inner product of $\langle \overrightarrow{X_A}, t\overrightarrow{w} \rangle$ is zero. For the repeated file user B, the first two steps are the same for the user B.

Then, he also gets the result of download right $\overrightarrow{X_B}$ and sends it to the cloud. Moreover, the inner product of $\langle \vec{w}, t\overrightarrow{X_B'} \rangle$ calculates the process as follows:

$$\begin{aligned} < \vec{w}, \overrightarrow{X_B'} > &= \vec{w} \cdot \left(x_{B,0}, x_{B,0}, \ldots, x_{B,n-3}', x_{B,n-2}', x_{B,n-1}' \right)^T \\ &= \sum_{i=0}^{n-4} w_i x_{B,i} + w_{n-3} x_{B,n-3}' + \left(w_{n-2} x_{B,n-2}' + w_{n-1} x_{B,n-1}' \right) \\ &= \sum_{i=0}^{n-4} w_i x_{B,i} + w_{n-3} x_{B,n-3}' + y \\ r &= \left[\left(\sum_{i=0}^{n-4} w_i x_{B,i} \right) + \left(w_{n-3} x_{B,n-3} + \left(y' - y \right) \right) \right] + y \\ &= \left[-y' + \left(y' - y \right) \right] + y = (-y) + y = 0. \end{aligned} \tag{20}$$

In a word, all legal users who hold the download right corresponding to file M_A can pass the PoW. □

8. Security Analysis

This part will prove that the deduplication label is indistinguishable except for duplication check process, and the downloading right is resistant to forgery. To begin with, the security about deduplication label is given in Theorem 3.

Theorem 3 (Security of deduplication label). *For legitimate users, whether uploading the same or different files to perform deduplication, the deduplication labels are only distinguishable to the duplication check process.*

Proof. The following analysis will be divided into two cases, with respect to the deduplication labels corresponding to same files and different files.

Case 1. Supposing user A and user B possess the same file. They have the same hash value $h_A = h_B$ of two identical files, and their deduplication labels are

$$\begin{aligned} \overrightarrow{v_A^{(1)}} &= \left(\mathrm{PSRG}(h_A) \cdot P + e_A^{(1)} \right) \cdot r_A^{(1)}, \\ \overrightarrow{v_B^{(1)}} &= \left(\mathrm{PSRG}(h_B) \cdot P + e_B^{(1)} \right) \cdot r_B^{(1)}. \end{aligned} \tag{21}$$

According to the deterministic algorithm $\mathrm{PSRG}(\cdot)$, we can see $\mathrm{PSRG}(h_A) = \mathrm{PSRG}(h_B)$. Moreover, for the common matrix P, it is obvious that $\mathrm{PSRG}(h_A) \cdot P = \mathrm{PSRG}(h_B) \cdot P$. However, e_A, e_B and r_A, r_B are randomly sampled from χ_q^m and $\{-1, 0, 1\}^m$, respectively. The probability that the deduplication labels are identical is $(1/(3q)^m) < (1/Q(m))$, which is negligible. Therefore, we claim that the results $\overrightarrow{v_A^{(1)}} = \overrightarrow{v_B^{(1)}}$ is almost impossible, which means $\overrightarrow{v_A^{(1)}}$ and $\overrightarrow{v_B^{(1)}}$ satisfy semantic security.

Case 2. Supposing user A and user B possess different files. That is to say, they have different file hash values that $h_A \neq h_B$, and the deduplication labels are

$$\begin{aligned} \overrightarrow{v_A^{(2)}} &= \left(\mathrm{PSRG}(h_A) \cdot P + e_A^{(2)} \right) \cdot r_A^{(2)}, \\ \overrightarrow{v_B^{(2)}} &= \left(\mathrm{PSRG}(h_B) \cdot P + e_B^{(2)} \right) \cdot r_B^{(2)}. \end{aligned} \tag{22}$$

Similarly, since $\mathrm{PSRG}(h_A) \neq \mathrm{PSRG}(h_B)$, the probability that deduplication labels are the same is $(1/(n+1)(3q)^m) < (1/Q(m))$, which is indistinguishable from the distribution of Case 1.

Therefore, we can conclude that, since the deduplication labels of the same file are different, Case1 is of the same distribution indistinguishable from Case2, and the deduplication labels are semantic secure. In summary, the deduplication tags corresponding to the same file and different files are indistinguishable. □

Theorem 4 (Security proof of downloading right). *None of the users can forge a valid downloading right $\overrightarrow{X_A}$ which can deceive access control.*

Specifically, the security analysis of the downloading right can be guided by Lemmas 1 and 2.

Lemma 1 *After the original uploader A outsourced the file M_A to the cloud, the entire download certificate $\overrightarrow{w}$ is known only by the cloud.*

Proof. According to inner product predicate, the user A's downloading right $\overrightarrow{X_A}$ can make the inner products $\langle \overrightarrow{X_A}, t\overrightarrow{w} \rangle$ output 0.

However, the download certificate $\overrightarrow{w}$ is calculated by the user A who samples $w_i (i = 0, \ldots, n-2)$ and sets the last element w_{n-1} of the download certificate $\overrightarrow{w}$ to be $w_{n-1} = -((\sum_{i=0}^{n-2} x_{A,i} w_i)/(x_{n-1})) \bmod q$.

Then, when the user A uploads for the first time, the cloud obtains the completed download certificate $\overrightarrow{w}$ corresponding to A's secret attributes. For now, if there is an illegal user who tries to falsify the download certificate $\overrightarrow{w} \xleftarrow{\$} \mathbb{Z}_q^n$ to cheat the PoW system, his advantage is

$$\Pr\left[< \overrightarrow{X_A}, \overrightarrow{w'} > = 0 \right] = \frac{1}{q^{(n-1)}} \leq \frac{1}{Q(n)}, \tag{23}$$

which is negligible. □

Lemma 2. *For repeated file uploaders, they do not know the remaining elements of the download certificate $\overrightarrow{w}$ except for $(w_{n-3}, w_{n-2}, w_{n-1})$.*

Proof. Take a repeated file uploader B as an example, he uses $(w_{n-3}, w_{n-2}, w_{n-1})$ to update the last three elements of download right $\overrightarrow{X_B}$ into $(x'_{B,n-3}, x'_{B,n-2}, x'_{B,n-1})$.

In detail,

$$x'_{B,n-3} = x_{B,n-3} - \frac{(y - y')}{w_{n-3}}, \tag{24}$$

$$w_{n-2} x'_{B,n-2} + w_{n-1} x'_{B,n-1} = y. \tag{25}$$

Since the value of w_{n-3} is known, the result of $x'_{B,n-3}$ can be calculated. However, because the rank of formula (25) is equal to 1 and $w_{n-2} x'_{B,n-2} + w_{n-1} x'_{B,n-1} = y$, the formula of (25) contains two unknowns variables. Thus, the results of $x'_{B,n-2}$ and $x'_{B,n-1}$ are infinite. Therefore, when the user B calculates the downloading right, he does not know the remaining elements of the download certificate $\overrightarrow{w}$ except for $(w_{n-3}, w_{n-2}, w_{n-1})$.

Considering that the solutions of formula (25) are infinite, the security of downloading right can be effectively protected, namely, $\overrightarrow{X_B}$ of the user B. Thus, it also guarantees the confidentiality of legal users' attributes. If an illegal user attempts to forge the remaining $n-3$ elements of $\overrightarrow{w}$ to get the new download certificate $\overrightarrow{w''} = (w''_0, \ldots, w''_{n-4}, w_{n-3} \ldots, w_{n-1})$, his advantage is just

$$\Pr\left[< \overrightarrow{X'_B}, \overrightarrow{w''} > = 0 \right] = \frac{1}{q^{(n-3)}} \leq \frac{1}{Q(n)}, \tag{26}$$

which is negligible. Therefore, our scheme will not expose the remaining elements of the download certificate $\overrightarrow{w}$.

In terms of Lemmas 1 and 2, it can be seemed that no user can forge a valid downloading right since the complete download certificate and the attributes vector $\overrightarrow{\mu_A}$ will not be exposed. □

9. Performance Analysis

Then, the performance of our schemes will be analysed comparing with other main technologies. The notation of symbols can be found in Table 1, as for functions, such as the necessity of third-party, deduplication level, participants, and the necessity of key fusion, and the comparison can be found in Table 2.

Compared with the schemes from [2, 3, 9], our scheme does not require any third-party, which effectively avoided extra trusting relationships and can save numerous computation/communication resources. Moreover, our scheme executes file deduplication amongst multiple users, implying that it is more flexible and more adaptive to various cloud environment. From the perspective of key fusion, when compared with the literature from [2, 3, 8, 9], any key fusion process is unnecessary in our scheme, so that it can be applied even if the user resources are limited.

Then, we compare the computation overheads for deduplication taken by the client, third-party, and cloud in the above schemes. The details are given in Table 3.

Compared with the cost on client side in scheme [3], that of our scheme is $O(f)$Hash + $O(1)$PSRG, where a pseudorandom number sequence is generated instead of N convergence keys. In fact, it means that our scheme is more efficient since PSRG can be iterated generated via small numbers, not saying that our scheme if free of any third-party. Moreover, the hash value of file can be secretly used as the encryption key in this paper. Therefore, there is no need for multiple users to reconstruct the convergence key, which further outperformed the scheme of [3] by avoiding the consumption of key distribution and fusion.

Compared with the schemes in [8, 9], our method does not need to construct Bloom filter or attribute binary tree on client side, so the computational cost is slightly advantageous. In addition, since our scheme does not involve any third-party, the computational cost of TTP can be neglected. As for the overhead on cloud side, our scheme does not have to initialize any ownership data structure compared with that of schemes [8, 9]. Therefore, the calculation is deduced to $O(g)$ since it is not related to the file size but only to the length hash value.

Now, we compare the computational overhead for PoW, respectively on client, third-party and cloud side. The results are shown in Table 4.

It can be seen from Table 4 that users have to preserve and search the Bloom filter or attribute binary tree to accomplish PoW in [2, 3, 8, 9]. So, there is an additional cost $O(kL)$ or $O(N \log N)$ on the client side. However, our scheme does not require this process, so the calculation cost is only $O(f)$Hash + $O(n)$Add, where the second term is just n times of add operation. Comparing the cost on cloud side, our scheme dose also outperformed that of [2, 3, 8, 9], which

User B

Input: $((w_{n-3}, w_{n-2}, w_{n-1}), y, \overrightarrow{\mu_B})$

(1) Samples $\overrightarrow{u_B} = (u_{B,0}, u_{B,n-1}, u_{B,n-2}, \ldots, u_{B,1})$, where $u_{B,i}$ is irreversible and cofficients belong to $\mathbb{Z}_q$

(2) Computes $f(\overrightarrow{\mu_B}) \leftarrow \overrightarrow{\mu_B}$, $g(\overrightarrow{u_B}) \leftarrow \overrightarrow{u_B}$, $x_{B,0} \leftarrow g(\overrightarrow{u_B}) \cdot f(\overrightarrow{\mu_B})$
for all $k \in \{1, \ldots, n-1\}$
$g(\overrightarrow{u_{B,k}}) \leftarrow g(\overrightarrow{u_B}) \cdot x \bmod x^n$, $x_{B,k} \leftarrow g(\overrightarrow{u_{B,k}}) \cdot f(\overrightarrow{\mu_B})$, $\overrightarrow{u_B} \leftarrow \overrightarrow{u_{B,k}}$, $k \leftarrow k+1$

(3) Computes $y' \leftarrow w_{n-2} \cdot x_{B,n-2} + w_{n-1} \cdot x_{B,n-1}$, $x_{B,n-3}' \leftarrow x_{B,n-3} - (y - y')/w_{n-3}$

(4) Samples $x_{B,n-2}' \xleftarrow{\$} \mathbb{Z}_q^n$, $x_{B,n-2} \leftarrow x_{B,n-2}'$; and computes $x_{B,n-1} \leftarrow ((y - w_{n-2} \cdot x_{B,n-2}')/(w_{n-1}))$

(5) For all $k \in \{1, \ldots, n-1\}$

(6) $u_{B,i}\mu_{B,0} + u_{B,[i+(n-1)] \bmod n}\mu_{B,1} + \cdots + u_{B,(i+1) \bmod n}\mu_{B,n-1} \leftarrow x_{B,i}$

Output: $\left\{\overrightarrow{u_B} = t(u_{B,n-1}, \ldots, u_{B,1}, u_{B,0})\right\}$

Algorithm 1: Calculation process chart of repeated file.

User A

Input: $(\overrightarrow{u_A}, \overrightarrow{\mu_A}, n, \mathrm{sk}_A)$

(1) Computes $f(\overrightarrow{\mu_A}) \leftarrow \overrightarrow{\mu_A}$, $g(\overrightarrow{u_A'}) \leftarrow g(\overrightarrow{u_A}) \cdot x \bmod x^n$, $\overrightarrow{u_A} \leftarrow u_A'$, $x_{A,0} \leftarrow g(\overrightarrow{u_A}) \cdot f(\overrightarrow{\mu_A})$

(2) Computes $f(\overrightarrow{\mu_B}) \leftarrow \overrightarrow{\mu_B}$, $g(\overrightarrow{u_B}) \leftarrow \overrightarrow{u_B}$;
$x_{B,0} \leftarrow g(\overrightarrow{u_B}) \cdot f(\overrightarrow{\mu_B})$

(3) For all $k \in \{1, \ldots, n-1\}$
$g(\overrightarrow{u_{A,k}}) \leftarrow g(\overrightarrow{u_A}) \cdot x \bmod x^n$, $x_{A,k} \leftarrow g(\overrightarrow{u_{A,k}}) \cdot f(\overrightarrow{\mu_A})$
$\overrightarrow{u_A} \leftarrow \overrightarrow{u_{A,k}}$, $k \leftarrow k+1$

(4) Sends $\left\{\overrightarrow{X_A} = (x_{A,0}, x_{A,1}, \ldots, x_{A,n-1})\right\}$ to Cloud.

Cloud

Input: $(\overrightarrow{w}, \overrightarrow{X_A}, C_A, \alpha, q)$

(5) Computes $<\overrightarrow{w}, \overrightarrow{X_A}> \leftarrow \sum_{i=0}^{n-1} w_i \cdot x_{A,i}$

(6) If $<\overrightarrow{w}, \overrightarrow{X_A}> = 0$
Output: $\{C_A, (w_{n-3}, w_{n-2}, w_{n-1}), y\}$; Otherwise
Output: NULL.

(7) Sends $\{C_A, (w_{n-3}, w_{n-2}, w_{n-1}), y\}$ to User A

Algorithm 2: Process chart of PoW for the original file user.

User A

Input: $(\overrightarrow{\mu_{\text{team}}}, n, (w_{n-3}, w_{n-2}, w_{n-1}), y)$

(1) Samples $\overrightarrow{u_{\text{team}}} = (u_{\text{team},0}, u_{\text{team},1}, \ldots, u_{\text{team},n-1})$, where $u_{\text{team},0}$ is irreversible with cofficients belong to $\mathbb{Z}_q$

(2) Computes $f(\overrightarrow{\mu_{\text{team}}}) \leftarrow \overrightarrow{\mu_{\text{team}}}$
for all $k \in \{1, \ldots, n-1\}$
$g(\overrightarrow{u_{\text{team},k}}) \leftarrow g(\overrightarrow{u_{\text{team}}}) \cdot x \bmod x^{n+1}$; $x_{\text{team},(n-k-1)} \leftarrow g(\overrightarrow{u_{\text{team},k}}) \cdot f(\overrightarrow{\mu_{\text{team}}})$; $\overrightarrow{u_{\text{team}}} \leftarrow \overrightarrow{u_{\text{team},k}}$; $k \leftarrow k+1$

(3) Computes $y' \leftarrow w_{n-2} \cdot x_{\text{team},n-2} + w_{n-1} \cdot x_{\text{team},n-1}$

(4) Computes $x'_{\text{team},n-3} \leftarrow x_{\text{team},n-3} - ((y - y')/(w_{n-3}))$

(5) Samples $x'_{\text{team},n-2} \xleftarrow{\$} \mathbb{Z}_q^n$, $x_{\text{team},n-2} \leftarrow x'_{\text{team},n-2}$, and Computes $x_{\text{team},n-1} \leftarrow (y - w_{n-2} \cdot x'_{\text{team},n-2})/w_{n-1}$

(6) For all $k \in \{1, \ldots, n-1\}$
$u_{\text{team},i} \cdot \mu_{\text{team},0} + u_{\text{team},[i+(n-1)] \bmod n} \cdot \mu_{\text{team},1} + \cdots + u_{\text{team},(i+1) \bmod n} \cdot \mu_{\text{team},n-1} \leftarrow x_{\text{team},i}$

Output: $\left\{\overrightarrow{u_{\text{team}}} = (u_{\text{team},0}, u_{\text{team},n-1}, \ldots, u_{\text{team},1})\right\}$

Algorithm 3: Calculation process chart of common downloading right.

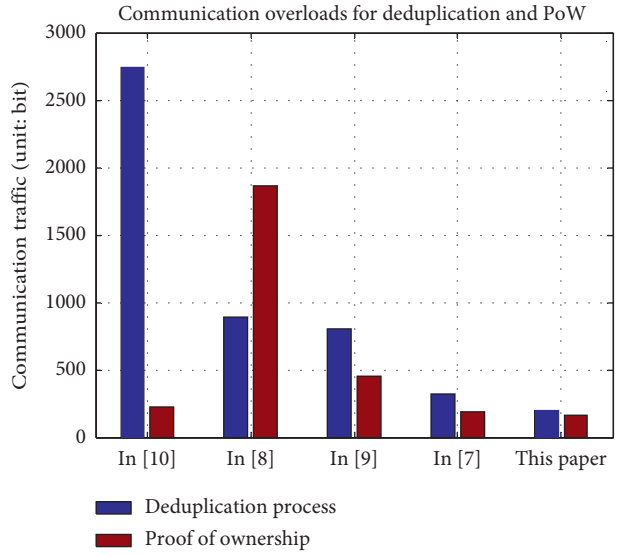

Figure 4: Histogram of communication cost of similar schemes.

Table 2: Function comparison between main data deduplication schemes.

Schemes	Technology	TTP	Level	Object	Key fusion
[8]	BL-MLE + PoW	—	Block	Single user	Yes
[3]	Threshold blind signature + verifiable secret sharing	Key servers	File	Single user	Yes
[2]	Authentication protocol + authorization detection	Cloud server	File	Multiple users	Yes
[9]	Attribute encryption + random sampling	Attribute center	Block	Multiple users	Yes
This paper	Attribute access policy + inner product predicate	—	File	Multiple users	No

Table 3: Computation overheads for deduplication.

Schemes	Client	TTP	Cloud
[8]	$O(b)$Hash · Hash	—	$O(b)$PoW
[3]	$O(f)$Hash + $O(N)$CE_K	$O(f)$Hash	$O(g)$
[2]	$O(f)$	$O(f)$	—
[9]	$O(f)$Hash + $O(f)$PoW	$O(f)$Hash	$O(f)$PoW
This paper	$O(f)$Hash + $O(1)$PSRG	—	$O(g)$

Table 4: Computation overheads for PoW.

Schemes	Client	TTP	Cloud
[8]	$O(b)$Hash + $O(b)$	—	$O(f)$Add
[3]	$O(f)$Hash + $O(kL)$	$O(N)$CE_K	$O(kLf)$Add
[2]	$O(f) + O(kL)$	—	$O(kLf)$Add
[9]	$O(f)$Hash + $O(N \log N)$	$O(N)$CE_K	$O(kLf)$Add
This paper	$O(f)$Hash + $O(n)$Add	—	$O(n)$Add

is $O(n)$Add. The reason is similar that the calculation cost on cloud side has nothing to do with the file size but only the number of attributes.

Finally, taking the file of 256 bits as an example, we compare the communication overhead for deduplication and PoW amongst the same set of schemes. The details are shown in Figure 4.

According to Figure 4, our scheme has obvious advantage on communication overheads compared with other schemes. Our solution can effectively reduce the usage of bandwidth as well as time delay. Moreover, since all deduplication check and ownership proof processes are independent, our scheme is capable of parallel processing, which is more fit for batch implementation.

10. Conclusions

This paper proposed a novel deduplication scheme based on LWE and FE to balance the conflict between the accessibility and the indistinguishability of data. Focusing on the purpose of deduplication check, LWE is exploited to construct deduplication labels which are distinguishable only if their deduplication certificates are known. To realize more efficient and flexible access control, inner product predicate is used that data can be retrieved only if both users downloading right and attributes vector are possessed. Thanks to the separation of downloading right and user's attributes, the downloading right can be recalculated for repeated uploading and authorization transfer without changing the corresponding deduplication label or download certificate over cloud. Correctness and security analyses proved that deduplication can be accomplished only by the duplication check process with negligible false positive, and it is almost impossible for any adversaries to fabricate a legal downloading right. Compared with other main technologies, our scheme is more applicable to multiuser environment and freed from trusted third-party. Since both duplication check and ownership proof are realized by inner product, the performances of computation and communication are more advantageous in our method, not mentioning its capacity of batch processing due to parallelism.

Data Availability

The data set was obtained from Chongqing Tongnan Electric Power Co., Ltd (telephone: 023-44559308; official website: http://www.12398.gov.cn/html/information/753078881/753078881201200006.shtml).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank Bo Mi and Fengtian Kuang for their comments and suggestions. This work was supported in part by the National Natural Science Foundation of P.R. China (Grant nos. 61573076, 61703063, and 61903053); the Science and Technology Research Project of the Chongqing Municipal Education Commission of P.R. China (Grant nos. KJZD-K201800701, KJ1705121, and KJ1705139); and the Program of Chongqing Innovation and Entrepreneurship for Returned Overseas Scholars of P.R. China (Grant no. cx2018110).

References

[1] M. Wen, K. Ota, H. Li, and J. Lei, "Secure data deduplication with reliable key management for dynamic updates in cpss," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 4, pp. 1–11, 2016.

[2] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou, "A hybrid cloud approach for secure authorized deduplication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1206–1216, 2015.

[3] M. Miao, J. Wang, H. Li, and X. Chen, "Secure multi-server-aided data deduplication in cloud computing," *Pervasive and Mobile Computing*, vol. 24, pp. 129–137, 2015.

[4] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: deduplication in cloud storage," *IEEE Security & Privacy Magazine*, vol. 8, no. 6, pp. 40–47, 2010.

[5] B. Mi, D. Huang, S. Wan, L. Mi, and J. Cao, "Oblivious transfer based on NTRUEncrypt," *IEEE ACCESS*, vol. 6, pp. 35283–35291, 2018.

[6] L. Li, X. Chen, X. Huang et al., "Secure distributed deduplication systems with improved reliability," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 356-357, 2015.

[7] P. Puzio, R. Molva, M. Önen et al., "ClouDedup: secure deduplication with encrypted data for cloud storage," *5th International Conference on Cloud Computing Technology and Science (Colud-Com)*, vol. 1, pp. 363–370, 2013.

[8] R. Chen, Y. Mu, G. Yang, et al.,F. Guo, "BL-MLE: block-level message-locked encryption for secure large file deduplication," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2643–2652, 2015.

[9] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: definitions and challenges," *Tcc*, vol. 2010, pp. 253–273, 2010.

[10] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," 2005.

[11] A. Zhou, S. Wang, S. Wan et al., "LMM: latency-aware microservice mashup in mobile edge computing environment," *Neural Computing and Applications*, vol. 4, no. 5, pp. 1–15, 2020.

[12] L. González-Manzano, J. M. D. Fuentes, and K. K. R. Choo, "Ase-PoW: a proof of ownership mechanism for cloud deduplication in hierarchical environments," 2016.

[13] Y. Chen, C. L. Li, J. L. Lan et al., "Secure sensitive data deduplication schemes based on deterministic/probabilistic proof of file ownership," *Journal on Communications*, vol. 36, no. 9, pp. 1–12, 2015.

[14] S. Wan, Y. Xia, L. Qi et al., "Automated colorization of a grayscale image with seed points propagation," *IEEE Transactions on Multimedia*, vol. 99, pp. 1–12, 2020.

[15] M. M. Xie, X. F. Liao, and Q. Zhou, "Generalized oblivious transfer protocol in distributed setting based on secret sharing," *Computer Engineering*, vol. 40, no. 3, pp. 184–187, 2014.

[16] S. Ding, S. Qu, Y. Xi, and S. Wan, "Stimulus-driven and concept-driven analysis for image caption generation," *Neurocomputing*, vol. 398, pp. 520–530, 2020.

[17] J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1615–1625, 2014.