electronics

MDPI

Article

Certificateless Remote Data Integrity Auditing with Access Control of Sensitive Information in Cloud Storage

Genqing Bian [1], Fan Zhang [1,*], Rong Li [2] and Bilin Shao [1]

[1] College of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710311, China
[2] Xi'an Aerospace Remots Sensing Data Technology Corporation, Xi'an 710000, China
* Correspondence: zhangfan@xauat.edu.cn; Tel.: +86-138-9204-5189

Abstract: With the spread of cloud storage technology, checking the integrity of data stored in the cloud effectively is increasingly becoming a concern. Following the introduction of the first remote data integrity audit schemes, different audit schemes with various characteristics have been proposed. However, most of the existing solutions have problems such as additional storage overhead and additional certificate burden. This paper proposes a certificateless remote data integrity auditing scheme which takes into account the storage burden and data privacy issues while ensuring the correctness of the data audit results. In addition, the certificateless design concept enables the scheme proposed in this paper to avoid a series of burdens brought by certificates. The scheme designed in this paper provides a data access control function whereby only users who hold a valid token generated by the data owner can access the target data from the cloud. Finally, this paper provides a detailed security proof to ensure the rationality of the results. A theoretical analysis and subsequent experimental verification show that the proposed scheme is both effective and feasible.

Keywords: cloud storage; integrity auditing; access control; digital signature

Citation: Bian, G.; Zhang, F.; Li, R.; Shao, B. Certificateless Remote Data Integrity Auditing with Access Control of Sensitive Information in Cloud Storage. *Electronics* **2022**, *11*, 3116. https://doi.org/10.3390/electronics11193116

Academic Editors: Alessandra De Benedictis and Roberto Nardone

Received: 31 August 2022
Accepted: 26 September 2022
Published: 29 September 2022

1. Introduction

With the development of the information revolution, human society has entered a highly informationized mode. The development of informatization has fundamentally changed the human way of life and brought great developments to the industrial society. As an important source of power for the development of social informatization, data are now been regarded as a strategic resource like oil and coal by many countries. Although the emergence of massive data has greatly promoted the progress of society, it has brought problems as well, for example, the extra cost of massive data storage. As data owners (DO) can spend a great deal of time and energy on local data storage, they generally choose to store these data using cloud service providers (CSP) with greater storage and computing capabilities. However, although this move reduces the burden of data ownership to a certain extent, it leads to the DO losing direct control over the data, that is, the owner may not be able to obtain the latest iteration of their data at any time. Data stored in the cloud can be corrupted for a variety of reasons. For example, in order to save on storage cost, a CSP may intentionally delete data that is not frequently accessed by the DO. In addition, malicious attacks from third parties or hardware failures on CSP's servers can cause data corruption. Furthermore, the CSP may choose to hide this from the DO in order to preserve their reputation. Such concealment by the CSP is likely to cause a degree of economic loss or other harm to the DO. In order to solve this problem, remote data integrity auditing scheme are increasingly being applied.

The first data integrity auditing scheme was proposed by Ateniese et al. [1] in 2007, in which the concept of provable data possession (PDP) was proposed. During the same period, Juels et al. [2] proposed another type of auditing scheme, proof of retrievability (PoR),

for the first time. Following these early developments, research on data integrity auditing schemes has made great progress, and many schemes with their own characteristics have been proposed one after the other [3–7].

In the auditing scheme developed and proposed here, the designer pays attention to realizing other properties in addition to ensuring the accuracy of data auditing, among which the most important is privacy protection. The privacy of data has become a topic of great concern in today's society. This is mainly because data often contain private information pertaining to the DO, which may cause serious harm if leaked. The existing solution is that, before uploading data to the cloud, the DO encrypts or blinds the data locally before generating tags and uploading [8,9]. Although this approach can effectively protect the privacy of data, encryption and blinding operations incur a huge computational burden on the DO, hindering the realization of the data's value.

In addition to privacy, data access control is a concern. Data can only be of maximum value when circulated. The most direct manifestation of this is the emergence of a large number of base stations in the edge environment. In order to realize the value of their data, the CSP or DO chooses to set up base stations in a fixed area and provides services to nearby users. Therefore, there are many data integrity auditing schemes in edge environments [10–12]. In the field of cloud storage, data access control cannot directly apply the model in the edge environment. This is because the cloud storage environment does not consider users in a fixed area, only authorized users who hold legal warrants to acquire data from the CSP. Therefore, data access control under cloud storage conditions must consider the correctness and revocability of warrants.

Finally, the core of the extant data integrity auditing schemes is digital signature technology. The DO guarantees the implementation of the scheme by generating corresponding tags for each data block. This approach can effectively check the integrity of data stored in the cloud, however, the appearance of tags means that the DO needs to prepare extra overhead to store tags in the cloud. In most of the existing schemes, the storage overhead generated by the tag is close to or even greater than the storage overhead of the data block itself [13–15]. The storage overhead incurred by these tags has become a major factor hindering the application of data auditing schemes.

1.1. Contribution

In order to solve the main problems with current auditing schemes, we design a certificateless remote data integrity auditing scheme with an access control function and sensitive information hiding. The main contributions of this paper are as follows:

(1) We propose a certificateless remote data integrity scheme that avoids the additional overhead brought by certificates. At the same time, in order to reduce the storage overhead caused by tags, our scheme does not choose to generate tags for each data block when generating tags for data blocks, instead aggregating multiple data blocks into one signature to reduce the number of tags.

(2) In order to make better use of the value of data, our scheme implements an access control function. That is, the DO generates warrants for authorized users, allowing them to acquire data from the cloud. The warrants genarated in our scheme are revocable, which means that DO can revoke a user's access at any time. At the same time, the scheme proposed in this paper takes privacy into account. During implementation, only authorized users, the DO, and the CSP can access the original master data, while unauthorized users and TPA cannot obtain the data by any means.

(3) We provide a detailed proof of the process of the proposed scheme and compare it with two other schemes through both theoretical analysis and experimental verification. The results show that the proposed scheme outperforms [8,15], mainly in terms of shorter time spent in tag generation and lower storage overhead on the part of the CSP.

1.2. Related Works

In 2012, Yang et al. [16] proposed a data integrity auditing scheme with privacy protection and support for dynamic updates, and which can support batch auditing for multiple DOs and CSPs. However, their approach is based on the traditional public key infrastructure design audit protocol, and as such there is a huge cost problem caused by the use of certificates. In 2013, Wang et al. [17] first proposed a certificateless public auditing scheme to verify the integrity of data in the cloud in order to solve the security risks brought on by public key infrastructure (PKI); however, they did not consider privacy or dynamic updating of data. In the same year, Kai et al. [18] proposed a data integrity auditing scheme in a multi-cloud environment with support for simultaneous verification of multiple audit requests for different data files stored on different CSPs by different DOs. The authors introduced a homomorphic encryption mechanism to ensure the privacy of the auditing process, however, the extra overhead caused by certificates was ignored. In 2014, Yuan et al. [19] proposed a remote data integrity auditing scheme with support for multi-user modification and collusion resistance based on full consideration of realistic scenarios. However, their approach did not address the risk of data leakage and was not able to guarantee privacy. In 2015, Jiang et al. [20] proposed a data integrity auditing scheme based on vector commitment and verifier-local revocation group signatures, however, the privacy of the data cannot be guaranteed by this approach. In 2016, Zhang et al. [21] proposed a lightweight auditing scheme based on entrusting most of the computation in the auditing process to the cloud in order to reduce the burden of auditing. Under this scheme third-party auditors can perform multiple audit tasks simultaneously, however, the risk of data leakage is increased. In 2017, Li et al. [22] designed a scheme to solve the complex key management problem in traditional methods by employing biometrics as a fuzzy identity; however, new computational overhead is generated in the process of key matching. In another paper from 2019, Shen et al. [5] designed an auditing scheme that does not require a private key using biometric data such as fingerprints as a fuzzy private key to sign data blocks to eliminate the storage burden of saving the private key. However, the feature of fuzzy private keys requires extra computational overhead to implement. In 2018, in order to better realize data sharing, Shen et al. [23] implemented data auditing and sharing by introducing a sanitizer to sanitize the data blocks containing sensitive information and their corresponding tags; however, the inclusion of these cleaning agents increases the computational cost of label generation. In 2020, Zhao et al. [24] embedded blockchain technology into the cloud auditing system and designed a data integrity auditing scheme with a privacy protection function by combining a blockchain and digital signature technology. However, the encryption algorithm they used to ensure data security adds an additional computational burden. In the same year, Lan et al. [25] pointed out that the RDIC protocol proposed by C. Sasikala et al. was not secure, and provided a rigorous proof analysis. In 2021, in order to avoid the waste of resources caused by repeatedly challenging specific data blocks over a short period of time, Wang et al. [26] confirmed the time at which such data blocks are checked by predicting user behavior and selecting the challenging data blocks on this basis. In 2022, Yang et al. [27] implemented a data integrity auditing scheme with support for dynamic insertion and provable seletion through a number-rank-based Merkle hash tree. In another 2022 paper, S. Dhelim et al. [28] proposed and designed a large-scale IoT trust management system able to effectively manage the trust relationships of devices in large-scale IoT contexts. Table 1 compares the characteristics of the works referenced above.

1.3. Organization

The rest of this paper is organized as follows. In Section 2, we describe several of the technologies that support our paper and then provide the scheme outline and security model. In Section 3, we elaborate our proposed remote data integrity auditing scheme. In Section 4, we analyze the security of our scheme in detail. Section 5 presents the results of our performance evaluation. Finally, we present the conclusions of this paper in Section 6.

Table 1. Reference scheme comparison table.

	Certificateless	Privacy	Dynamic Update	Data Sharing
[5]	✓	✓		
[16]		✓	✓	
[17]	✓			
[18]		✓		
[19]				✓
[20]			✓	✓
[21]			✓	
[22]		✓		
[23]		✓		✓
[24]		✓		
[26]				

2. Preliminaries

In this section, we provide relevant knowledge points and describe the security model of the scheme we designed. Important mathematical notations used in this paper are listed in Table 2.

Table 2. Mathematical notation.

Notation	Description
G_1, G_2	two multiplicative cyclic groups
q	a large prime number
g,	a generator of the multiplicative group G_1
e,	bilinear map
H_1, H_2, H_3,	three secure cryptographic hash functions
$u_1, u_2, \ldots, u_t$,	t distinct elements in group G_1
π	a pseudo-random permutation
φ	a pseudo-random function
σ_i	a signal tag
Φ	collection of tags
$chal$	a challenge message
P	a proof message

2.1. System Model

There are five entities in our scheme: DO, CSP, KGC, TPA, and Users. The relationship between them is shown in Figure 1.

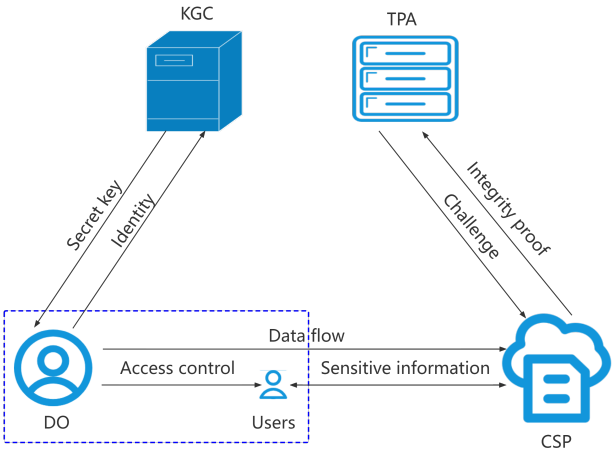

Figure 1. System model.

(1) KGC: The KGC is responsible for generating relevant parameters and generating a partial private key for the DO.

(2) DO: The DO is the actual owner of the data and is responsible for splitting the file into appropriate sizes, generating corresponding tags, and generating warrants for authorized users.

(3) CSP: The CSP is an institution with sufficient storage and computing power, and is responsible for properly storing data and making it available to authorized users.

(4) TPA: The TPA is an institution with sufficient computing power, and is responsible for randomly issuing data integrity challenges to the CSP. In our scheme, the TPA is considered semi-honest, i.e., while it honestly reports auditing results, it is curious about the data.

(5) Users: Users are the groups who want to use the data stored by the CSP; they can obtain data from the CSP using legitimate warrants.

2.2. Bilinear Maps

Let q be a large prime and G_1 and G_2 be two cyclic multiplicative groups with the same order q, where g is a generator of G_1 and $e : G_1 \times G_1 \to G_2$ is a bilinear map with the following properties:

(1) Bilinearity: for $\forall a, b \in G_1$ and $\forall x, y \in Z_q^*$, $e(a^x, b^y) = e(a, b)^{xy}$.

(2) Non-degeneracy: $\exists a, b \in G_1$, and $e(a, b) \neq 1_{G_2}$.

(3) Computability: for $\forall a, b \in G_1$, there is an efficient algorithm to calculate $e(a, b)$.

2.3. Security Assumptions

2.3.1. Computational Diffie–Hellman (CDH) Problem

Let G_1 be a multiplicative cyclic group, where g is a generator of G_1. Given the tuple $\left(g, g^a, g^b\right)$, where $a, b \in Z_q^*$ is unknown, the CDH problem is to calculate g^{ab}.

2.3.2. CDH Assumption

For a probabilistic polynomial time (PPT) adversary A, the advantage for A in solving the CDH problem in G_1 is negligible. Assume that ε is a negligible value; then, it can be defined as

$$Adv_{G_1 A}^{CDH} = \Pr\left[A\left(g, g^a, g^b\right) = g^{ab} : a, b \overset{R}{\leftarrow} Z_q^*\right] \leq \varepsilon$$

2.3.3. Discrete Logarithm (DL) Problem

Let G_1 be a multiplicative cyclic group, where g is a generator of G_1. Given the tuple (g, g^x), where $x \in Z_q^*$ is unknown, the DL problem is to calculate x.

2.3.4. DL Assumption

For a PPT adversary A, the advantage for A in solving the DL problem in G_1 is negligible. Assume that ε is a negligible value; then, it can be defined as

$$Adv_{G_1 A}^{DL} = \Pr\left[A(g, g^x) = x : x \overset{R}{\leftarrow} Z_q^*\right] \leq \varepsilon$$

2.4. Outline of Our Scheme

Our designed certificateless data integrity auditing scheme with sensitive information hiding involves eight algorithms with the following functions:

$Setup(1^k) \longrightarrow (params, msk)$: this algorithm is performed by the KGC to generate the relevant parameters and master key.

$Extract(params, ID, msk) \longrightarrow sk_1$: this algorithm is performed by the KGC, mainly to generate the partial private key for the DO.

$KeyGen(params, ID, sk_1) \longrightarrow (sk, pk)$: this algorithm is run by the DO, and its main function is to generate a complete private key.

$TagGen(params, sk, F, fname) \longrightarrow \Phi$: this algorithm is performed by the DO; its main function is to generate tags for each data block.

$Challenge(params, c) \longrightarrow chal$: this algorithm is executed by the TPA, and mainly generates relevant parameters used to challenge the integrity of the data blocks stored in the cloud.

$ProofGen(params, chal, F, \Phi) \longrightarrow P$: this algorithm is executed by the CSP to generate relevant proofs in response to TPA challenges.

$ProofVerify(params, pk, P, chal, ID) \longrightarrow \{0, 1\}$: this algorithm is executed by TPA, mainly to judge whether the proof of the CSP response is legal and to judge the integrity of the data.

$warrantGen(params, ID, ID^{'}) \longrightarrow W$: this algorithm is executed by the DO, mainly to generate warrants for authorized users for access control.

2.5. Security Model

The security model of our scheme is based on a game played between a challenger β and an adversary A, where the challenger β represents the DO and TPA and the adversary A represents the malicious cloud CSP. The specific details of the game are as follows:

Setup: The challenger β executes the $Setup$ algorithm to generate the master key msk and the public parameters $params$, then sends the public parameters to the adversary A and stores the msk properly.

Hash Queries: The adversary A can query any type of hash function, then challenger β performs related calculations and sends the results to A.

Private/Public Key Queries: The adversary A can query the private key of the user with identity ID. When receiving a private key query request from adversary A, challenger β runs $Extract$ and $KeyGen$, then feeds back the public key pk_{ID} and the combined private key sk_{ID} to A.

Tag Queries: The adversary A can randomly select a data block m_i and query its corresponding tag; the challenger β then executes $TagGen$ to generate the signature of the data block m_i and sends the result to A.

Integrity Proof Queries: The adversary A can randomly select any number of data blocks and query their corresponding integrity proof; the challenger β then executes $ProofGen$ and returns the result to A.

Challenge: The challenger β selects a certain number of data blocks to send to the adversary A in order to obtain a data integrity proof.

Forging: In response to a challenge message sent by challenger β, adversary A forges an integrity proof and returns it to β to verify the validity of the proof. If this proof can always pass the challenger's verfication, the adversary A is considered to have won the game.

Based on the above game, we have the following definitions.

Definition 1. *A certificateless remote data integrity auditing scheme is considered to be secure if the probability of adversary A winning the game against challenger β is negligible.*

Furthermore, we propose a formal definition to satisfy the security requirement of sensitive information hiding.

Definition 2. *A certificateless remote data integrity auditing scheme is regarded as having sensitive information hiding if only the DO and authorized users can obtain the original data from the CSP during the execution of the scheme.*

3. Our Proposed Scheme

In this part, we elaborate the details of our design scheme.

$Setup(1^k) \longrightarrow (params, msk)$: This algorithm is performed by the KGC. Given a security parameter k, the KGC randomly generates a large prime q with the feature

$|q| = k$. Then, the KGC generates two multiplicative cyclic groups G_1 and G_2, both of order q, where g is a generator of G_1 and $(u_1, u_2, \ldots, u_t) \in (G_1)^t$ represents t random elements. In addition, the KGC selects three secure cryptographic hash functions $H_1 : \{0,1\}^* \longrightarrow G_1, H_2 : \{0,1\}^* \times \{0,1\}^* \times G_1 \longrightarrow G_1, H_3 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \longrightarrow G_1$. Here, $\pi : Z_q^* \times \{1,2,\ldots,n\} \longrightarrow \{1,2,\ldots,n\}$ is a pseudo-random permutation (PRP), $\varphi : Z_q^* \times Z_q^* \longrightarrow Z_q^*$ is a pseudo-random function (PRF), e is a bilinear map acting on G_1, and G_2: $G_1 \times G_1 \longrightarrow G_2$. Then, the KGC generates the master secret key $msk = s$, where s is randomly selected from Z_q^* and the master public key mpk is calculated by $mpk = g^s$. After carrying out the above operations, the KGC publishes $params = (G_1, G_2, g, q, u_1, u_2, \ldots, u_t, e, mpk, H_1, H_2, H_3, \pi, \varphi)$ and keeps the msk private.

$Extract(params, ID, msk) \longrightarrow sk_1$: this algorithm is responsible for generating part of the private key for the DO. After receiving the ID of the DO, the KGC computes $sk_1 = H_1(ID)^{msk}$ and feeds the result back to the DO.

$KeyGen(params, ID, sk_1) \longrightarrow (sk, pk)$: when the DO receives the part of the private key sk_1 sent by the KGC, he or she first verifies whether $e(H_1(ID), mpk) = e(sk_1, g)$ is established. If the equation does not hold, the DO refuses to recognize the legitimacy of sk_1 and initiates a new round of application to the KGC. Otherwise, the DO randomly selects an element $sk_2 = x \in Z_q^*$ and computes $pk = g^x$. After completing the above operation, the DO publishes the public key pk and keeps the private key $sk = (sk_1, sk_2)$ secret.

$TagGen(params, sk, F, fname) \longrightarrow \Phi$: here, we assume that file F has been divided into n parts of $m_1, m_2, \ldots, m_n$ before uploading it to the cloud and that each part contains t small sectors with the same length, that is, $F = (m_{ij})_{n \times t}$, where $m_{ij} \in Z_q^*$ $(1 \le i \le n, 1 \le j \le t)$. $fname \in \{0,1\}^*$ is the unique ID of the file F. The DO computes the tag for each row block $(m_1, m_2, \ldots, m_n)$ by computing Equation (1):

$$\sigma_i = sk_1 \cdot \left(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}} \right)^{sk_2} \tag{1}$$

The set of all tags is denoted by $\Phi = \{\sigma_i | i \in \{1,2,\ldots,n\}\}$. After generating tags for all data row blocks, the DO transmits Φ to the CSP. When receiving the tag set Φ, the CSP can verify the single tag through $e(\sigma_i, g) = e(H_1(ID), mpk) \cdot e(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}}, pk)$.

$Challenge(params, c) \longrightarrow chal$: in order to reduce overhead, the TPA generally randomly selects partial data blocks when checking the integrity of data stored in the cloud. Suppose the TPA chooses to check c data blocks; then, the TPA randomly selects two elements $(k_1, k_2) \in (Z_q^*)^2$. Finally, the TPA sends the challenge message $chal = (c, k_1, k_2)$ to the CSP.

$ProofGen(params, chal, F, \Phi) \longrightarrow P$: after receiving the challenge request initiated by the TPA, the CSP first genarates the relevant parameters set $C = \{(\alpha_l, \beta_l) | l \in \{1,2,\ldots,c\}\}$, where $\alpha_i = \varphi(k_1, i), \beta_i = \pi(k_2, i)$. Then, the CSP selects a value $r \in Z_q^*$ and computes

$$M = \left\{ M_j | M_j = \sum_{i \in c} \alpha_i m_{\beta_i j} + r, j \in \{1,2,\ldots,t\} \right\}$$
$$\sigma = \prod_{i \in c} \sigma_{\beta_i}^{\alpha_i},$$
$$R = \prod_{j=1}^{t} u_j^r$$

After completing the above calculation, the CSP sends the proof $P = \{M, \sigma, R\}$ to the TPA.

$ProofVerify(params, pk, P, chal, ID) \longrightarrow \{0,1\}$: The TPA uses PRF and PRP locally to calculate the relevant parameter set $C = \{(\alpha_i, \beta_i)|i \in \{1,2,\ldots,c\}\}$. In this way, the TPA can conduct verification using Equation (2):

$$\begin{aligned} &e(\sigma, g) \cdot e(R, pk) = e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot \\ &e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j}, pk) \end{aligned} \tag{2}$$

If Equation (2) holds, the algorithm outputs a value of 1, indicating that the checked data is securely stored in the cloud. Otherwise, the data have been polluted, and the algorithm outputs 0.

$warrantGen(params, ID, ID^{'}) \longrightarrow W$: To generate a warrant that can access the data stored in the cloud, the DO and user $U_{ID'}$ randomly select $k_{ID}, k_{ID'} \in Z_q^*$, respectively. After completing the above operation, the DO and $U_{ID'}$ keep $k_{ID}, k_{ID'}$ private, and publish $g^{k_{ID}}, g^{k_{ID'}}, \alpha = (g^{k_{ID}})^{k_{ID}^{'}} = (g^{k_{ID}^{'}})^{k_{ID}}$, respectively. To authorize $U_{ID'}$ for sensitive data access, the DO computes a warrant $W = H_2(ID||ID^{'}||\alpha)^{k_{ID}}$ and sends it to $U_{ID'}$. After $U_{ID'}$ receives W, he or she calculates $H_2(ID||ID^{'}||\alpha)^{k_{ID}^{'}} \cdot W$ and sends it to the CSP to access the data. When the CSP receives the request, it checks whether

$$\begin{aligned} &e(H_2(ID||ID^{'}||\alpha)^{k_{ID}^{'}} \cdot W, g) \\ &= e(H_2(ID||ID^{'}||\alpha), g^{k_{ID'}}) \cdot e(H_2(ID||ID^{'}||\alpha), g^{k_{ID}}) \end{aligned}$$

holds or not. If the equation holds, then the CSP transmits the data to U_{ID}; otherwise, the CSP rejects the application. When the DO wants to revoke $U_{ID'}$'s warrant, he or she simply re-selects a new $k_{ID}^* \in Z_q^*$ and updates $g^{k_{ID}^*}$, meaning that $U_{ID'}$ can no longer obtain data from the CSP.

4. Security Proof

In this section, we provide a theoretical analysis of our designed scheme from four aspects: correctness, soundness, sensitive information hiding, and detectability.

4.1. Correctness Proof

The correctness of our proposed scheme can be summarized in four points: (1) if the partial private key sent by the KGC to the DO is correct, the DO always accepts it; (2) a single tag always passes validation if the DO generates tags correctly for the data blocks; (3) if the CSP stores the data properly in the cloud, every generated proof can be verified by the TPA; (4) if the user holds a valid warrant from the DO for authorization, the user can obtain sensitive data from the CSP.

Proof. (1) The correctness of the partial private key generated by the KGC for the DO is as follows:

$$\begin{aligned} &e(H_1(ID), mpk) = e(H_1(ID), g^s) \\ &= e(H_1(ID)^s, g) = e(sk_1, g) \end{aligned}$$

(2) A single tag can be generated by Equation (1), and its correctness is proven as follows:

$$
\begin{aligned}
& e(\sigma_i, g) \\
&= e(sk_1 \cdot \left(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}} \right)^{sk_2}, g) \\
&= e(sk_1, g) \cdot e(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}}, g^{sk_2}) \\
&= e(H_1(ID), g^s) \cdot e(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}}, g^x) \\
&= e(H_1(ID), mpk) \cdot e(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}}, pk)
\end{aligned}
$$

(3) The correctness of auditing relies on checking Equation (2), the correctness of which is proven as follows:

$$
\begin{aligned}
& e(\sigma, g) \cdot e(R, pk) \\
&= e(\prod_{i \in c} \sigma_{\beta_i}^{\alpha_i}, g) \cdot e(R, pk) \\
&= e(\prod_{i \in c} (sk_1 \cdot \left(H_3(ID||fname||\beta_i) \cdot \prod_{j=1}^{t} u_j^{m_{\beta_i j}} \right)^{sk_2})^{\alpha_i}, g) \cdot \\
& e(R, pk) \\
&= e(\prod_{i \in c} sk_1^{\alpha_i}, g) \cdot e(R, pk) \cdot \\
& e(\prod_{i \in c} \left(H_3(ID||fname||\beta_i) \cdot \prod_{j=1}^{t} u_j^{m_{\beta_i j}} \right)^{\alpha_i}, g^{sk_2}) \\
&= e(\prod_{i \in c} H_1(ID)^{\alpha_i}, g^s) \cdot e(R, pk) \cdot \\
& e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{i \in c} \prod_{j=1}^{t} u_j^{\alpha_i \cdot m_{\beta_i j}}, g^x) \\
&= e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot e(\prod_{j=1}^{t} u_j^r, g^x) \cdot \\
& e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{i \in c} \prod_{j=1}^{t} u_j^{\alpha_i \cdot m_{\beta_i j}}, g^x) \\
&= e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot \\
& e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{\sum_{i \in c} \alpha_i \cdot m_{\beta_i j} + r}, g^x) \\
&= e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot \\
& e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j}, pk)
\end{aligned}
$$

(4) If the user holds the correct warrant, then it can be verified by the CSP as follows:

$$\begin{aligned} & e(H_2(ID||ID^{'}||\alpha)^{k_{ID}^{'}} \cdot W, g) \\ & = e(H_2(ID||ID^{'}||\alpha)^{k_{ID}^{'}}, g) \cdot e(W, g) \\ & = e(H_2(ID||ID^{'}||\alpha), g^{k_{ID}^{'}}) \cdot e(H_2(ID||ID^{'}||\alpha), g^{k_{ID}}) \end{aligned}$$

□

4.2. Soundness Proof

Theorem 1 (*Auditing soundness*)**.** *If the CDH assumption holds on the multiplicative cycle groups G_1 and G_2, then a malicious cloud cannot forge a verifiable proof.*

Proof. We prove through a series of games that there exists an extractor to solve the CDH problem by extracting the interaction information between the challenger β and adversary A if the CSP has forged a reasonable proof P^* without saving the complete data. □

Game 0. The specific process of this game has been elaborated in Section 2, therefore, we omit it here.

Game 1. This game is largely the same as *Game* 0, except with the addition of a new rule. Challenger β maintains a list locally to record Tag Queries made by adversary A. If adversary A generates a proof P that is successfully verified by the challenger β and the proof contains at least one tag that is not recorded in the list, then challenger β terminates the game and declares defeat.

Analysis. Assuming that adversary A wins *Game* 1 with a non-negligible probability, then there is an extractor that can solve the CDH problem through the following steps. Without loss of generality, suppose that the identity of the adversary is ID, the file is $F = (m_{ij})_{n \times t}$, and the file name is $fname$.

Suppose the honest prover produces the correct proof $P = \{M, \sigma, R\}$; then, it has

$$\begin{aligned} & e(\sigma, g) \cdot e(R, pk) = e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot \\ & e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j}, pk) \end{aligned} \tag{3}$$

When adversary A successfully forges a verifiable proof $P^* = \{M^*, \sigma^*, R\}$, it has

$$\begin{aligned} & e(\sigma^*, g) \cdot e(R, pk) = e(H_1(ID)^{\sum_{i=1}^{c} \alpha_i}, mpk) \cdot \\ & e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j^*}, pk) \end{aligned} \tag{4}$$

Obviously, $M \neq M^*$; otherwise, $\sigma = \sigma^*$. Given $g, g^{\alpha}, h \in G_1$, its goal is to compute h^{α}. The extractor randomly selects $r_i \in Z_q^*$ for each $i (1 \leq i \leq c)$ in the challenge, then sets $u_j = (g^a)^{\beta_j} \cdot (h^b)^{\gamma_j}$, where $a, b, \beta_j, \gamma_j \in Z_q^*$ and $1 \leq j \leq t$. This means that $\prod_{j=1}^{t} u_j^{m_{ij}} = \prod_{j=1}^{t} [(g^a)^{\beta_j} \cdot (h^b)^{\gamma_j}]^{m_{ij}} = (g^a)^{\sum_{j=1}^{t} \beta_j m_{ij}} \cdot (h^b)^{\sum_{j=1}^{t} \gamma_j m_{ij}}$. Meanwhile, the extractor randomly selects an element $s \in Z_q^*$ as msk and then performs *Extract* and *KeyGen* to generate the private key $sk = (sk_1, sk_2) = (H_1(ID)^s, x)$. Then, the extractor randomly selects an element $k \in Z_q^*$ and sets $pk = g^x = (g^{\alpha})^k$, which means that $x = k \cdot \alpha$. Finally, for each i, the extractor sets

$$H_3(ID||fname||i) = g^{r_i} / (g^a)^{\sum_{j=1}^{t} \beta_j m_{ij}} \cdot (h^b)^{\sum_{j=1}^{t} \gamma_j m_{ij}}$$

Hence, the extractor can compute $\sigma_i = sk_1 \cdot \left(H_3(ID||fname||i) \cdot \prod_{j=1}^{t} u_j^{m_{ij}} \right)^{sk_2} = H_1(ID)^s \cdot (g^{r_i})^x$.

Thus, by dividing Equations (4) and (3) we can obtain

$$\begin{aligned} & e(\sigma^*/\sigma, g) \\ & = e(\prod_{j=1}^{t} u_j^{M_j^*-M_j}, (g^\alpha)^k) \\ & = e(\prod_{j=1}^{t}((g^a)^{\beta_j} \cdot (h^b)^{\gamma_j})^{M_j^*-M_j}, (g^\alpha)^k) \\ & = e((g^a)^{\sum_{j=1}^{t}\beta_j \cdot (M_j^*-M_j)} \cdot (h^b)^{\sum_{j=1}^{t}\gamma_j \cdot (M_j^*-M_j)}, (g^\alpha)^k) \\ & = e(g^{a \cdot k \cdot \sum_{j=1}^{t}\beta_j \cdot \triangle M_j} \cdot h^{b \cdot k \cdot \sum_{j=1}^{t}\gamma_j \cdot \triangle M_j}, g^\alpha) \end{aligned} \tag{5}$$

From Equation (5), we can obtain

$$h^\alpha = (\sigma^* \cdot \sigma^{-1} \cdot g^{-a \cdot k \cdot \sum_{j=1}^{t}\beta_j \cdot \triangle M_j})^{1/(b \cdot k \cdot \sum_{j=1}^{t}\gamma_j \cdot \triangle M_j)}$$

According to the above equation, the CDH problem is unsolvable only when $b \cdot k \cdot \sum_{j=1}^{t}\gamma_j \cdot \triangle M_j = 0 \ mod \ q$. Obviously, in the finite field generated by the large prime number q, the probability of $b \cdot k \cdot \sum_{j=1}^{t}\gamma_j \cdot \triangle M_j = 0$ is infinitely close to $1/q$. This means that if adversary A successfully forges a verifiable proof, the extractor can solve the CDH problem with probability $1 - 1/q$, which contradicts the difficulty of solving the CDH problem.

Game 2. *Game* 2 introducesa new rule on the basis of *Game* 1, that is, if the challenger β finds that M^* in proof P^* generated by adversary A is not equal to the expected M, then the challenger terminates the game and declares defeat.

Analysis. Given $g, h \in G_1$, we build an extractor the purpose of which is to compute a value $x \in Z_q^*$ which satisfies $h = g^x$. We set $u_j = (g^a)^{\beta_j} \cdot (h^b)^{\gamma_j}$, where $a, b, \beta_j, \gamma_j \in Z_q^*$ and $1 \leq j \leq t$.

Suppose P is a correct proof generated by the honest cloud; then, we have

$$\begin{aligned} & e(\sigma, g) \cdot e(R, pk) = e(H_1(ID)^{\sum_{i=1}^{c}\alpha_i}, mpk) \cdot \\ & e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j}, pk) \end{aligned}$$

The adversary then generates a proof P^*, meaning that we have

$$\begin{aligned} & e(\sigma^*, g) \cdot e(R, pk) = e(H_1(ID)^{\sum_{i=1}^{c}\alpha_i}, mpk) \cdot \\ & e(\prod_{i \in c} H_3(ID||fname||\beta_i)^{\alpha_i} \cdot \prod_{j=1}^{t} u_j^{M_j^*}, pk) \end{aligned}$$

From *Game* 1, we know that $\sigma^* = \sigma$. Thus, the extractor has

$$\prod_{j=1}^{t} u_j^{M_j} = \prod_{j=1}^{t} u_j^{M_j^*}$$

It is clear that

$$\begin{aligned} 1 & = \prod_{j=1}^{t} u_j^{M_j^*-M_j} = \prod_{j=1}^{t} u_j^{\triangle M_j} \\ & = (g^a)^{\sum_{j=1}^{t}\beta_j \triangle M_j} \cdot (h^b)^{\sum_{j=1}^{t}\gamma_j \triangle M_j} \end{aligned}$$

Therefore, the extractor can solve the DL problem as follows:

$$h = g^{-(a \cdot \sum_{j=1}^{t}\beta_j \triangle M_j)/(b \cdot \sum_{j=1}^{t}\gamma_j \triangle M_j)}$$

According to the above equation, the DL problem is unsolvable only when $a \cdot \sum_{j=1}^{t} \beta_j \triangle M_j = 0 \ mod \ q$. Obviously, in the finite field generated by the large prime number q, the probability of $a \cdot \sum_{j=1}^{t} \beta_j \triangle M_j = 0$ is infinitely close to $1/q$. This means that if adversary A successfully forges a verifiable proof, the extractor can solve the DL problem with probability $1 - 1/q$, which contradicts the difficulty of solving the DL problem. From the above analysis, it can be seen that the proposed scheme is secure unless the adversary successfully solves both the CDH problem and the DL problem.

4.3. Sensitive Information Hiding Proof

Theorem 2 (*Sensitive information hiding*)**.** *In our designed scheme, only the CSP, DO, and legitimate users authorized by the DO can access the original data.*

Proof. We demonstrate this proof in two parts: (1) the TPA cannot obtain the original data during the auditing process and (2) the DO-generated warrants for authorized users are hard to forge. □

In the process of auditing, the TPA can select several data blocks for multiple challenges; its purpose is to try to reverse the original data blocks through a sufficient number of polynomials. In our scheme, the proof generated by the CSP adds a blinding factor r to all linearly aggregated data blocks $M = \{M_j | M_j = \sum_{i \in c} \alpha_i m_{\beta_i j} + r, j \in \{1, 2, \ldots, t\}\}$, meaning that unless the TPA can solve r from $\prod_{j=1}^{t} u_j^r$, it cannot recover the data. However, due to the difficulty of solving the DL problem, it is very unlikely that the TPA will be able to recover the data through multiple challenges. In addition, unauthorized users and users who have revoked warrants and want to continue to obtain data have difficulty forging legitimate warrants. The reason for this is simple: if the warrant is successfully forged, it means that k_{ID} is solved from $g^{k_{ID}}$, which contradicts the difficulty of solving the DL problem. Therefore, as long as the DL assumption is always true, the scheme designed in this paper can guarantee the privacy of data.

5. Performance Analysis

In order to further analyze the performance of our scheme, in this section we compare our scheme with [8,15] from two aspects, namely, theoretical analysis and experimental verification. Figure 2 shows the mapping used for file segmentation and label generation in this paper.

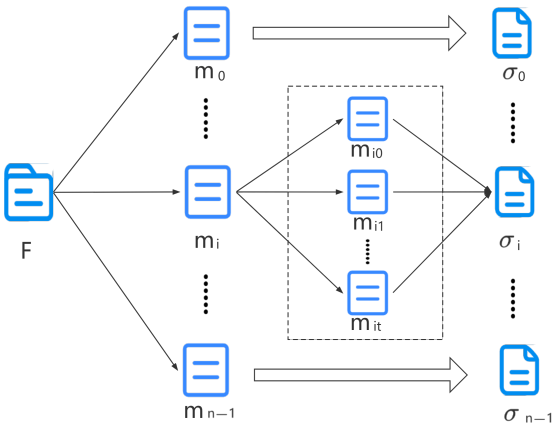

Figure 2. Mapping of data blocks and tags.

5.1. Theoretical Analysis

In this theoretical analysis, we analyze the three schemes from three aspects: *Storage cost*, *Communication cost*, and *Computation cost*. Through the analysis of relevant algorithms,

it is easy to see that *TagGen* is the core algorithm in the auditing scheme, and *Challenge*, *ProofGen* and *ProofVerify* are the algorithms with the highest execution frequency. Thus, we focus on these three algorithms and ignore the others. Before starting the comparison, we assume that file F is split into nt small chunks. The number of data blocks in each challenge is c.

5.1.1. Storage Cost

In analyzing the storage cost, we focus on the storage burden of the DO and CSP. In our scheme, the DO only needs to store his or her own private key; because the private key consists of two parts, the storage cost of the DO is $|G_1| + |Z_q^*|$. In addition, the CSP only needs to store the DO's data block and the corresponding tags locally. In our scheme, t data blocks correspond to one tag, meaning that the actual storage cost of the CSP is $nt|Z_q^*| + n|G_1|$. Table 3 lists the storage cost comparison of the three schemes in detail. It is obvious that our scheme greatly reduces the storage burden on the CSP.

Table 3. Storage cost.

	DO	**CSP**
Our Scheme	$\|G_1\| + \|Z_q^*\|$	$nt\|Z_q^*\| + n\|G_1\|$
[8]	$\|G_1\| + \|Z_q^*\|$	$nt\|G_1\| + nt\|Z_q^*\|$
[15]	$\|G_1\| + \|Z_q^*\|$	$nt\|G_1\| + nt\|Z_q^*\|$

5.1.2. Communication Cost

Communication cost is divided into three main stages. First, the DO transmits data blocks and tags to the CSP. In this process, the communication cost of our scheme is $nt|Z_q^*| + n|G_1|$. The second is the challenge request sent by the TPA to the CSP in the challenge stages. The communication cost of our scheme during this period is $3|Z_q^*|$. Finally, the CSP feeds the proof back to the TPA, for which the communication cost is $t|Z_q^*| + 2|G_1|$. Table 4 lists the storage cost comparison of the three schemes in detail.

Table 4. Communication cost.

	DO to CSP	**TPA to CSP**	**CSP to TPA**
Our Scheme	$nt\|Z_q^*\| + n\|G_1\|$	$3\|Z_q^*\|$	$t\|Z_q^*\| + 2\|G_1\|$
[8]	$nt\|G_1\| + nt\|Z_q^*\|$	$2c\|Z_q^*\|$	$\|Z_q^*\| + 2\|G_1\|$
[15]	$nt\|G_1\| + nt\|Z_q^*\|$	$3\|Z_q^*\|$	$\|Z_q^*\| + 4\|G_1\|$

5.1.3. Computation Cost

Before we start our analysis, we audit T_p for the computation of one time bilinear mapping, T_e for the computation of one time exponentiation operation, and T_m for the computation of one time multiplication. In the tag generation stage, the total computation of our scheme is $n(t+2)T_m + n(t+1)T_e$. In the proof generation stage, the total computation of our scheme is $(ct+c+t)T_m + (c+t)T_e$. In the proof verification stage, the total computation of our scheme is $4T_p + (2+c+t)T_m + (1+c+t)T_e$. Table 5 lists the computation cost comparison of the three schemes in detail. It can be seen from Table 5 that the advantage of our scheme is reflected in a low computation cost in the tag generation stage.

Table 5. Computational cost.

	TagGen	**ProofGen**	**ProofVerify**
Our Scheme	$n(t+2)T_m + n(t+1)T_e$	$(ct+c+t)T_m + (c+t)T_e$	$4T_p + (2+c+t)T_m + (1+c+t)T_e$
[8]	$2nt(T_m + T_e)$	$(2c+1)T_m + cT_e$	$3T_p + (c+1)T_m + (c+2)T_e$
[15]	$2nt(T_m + T_e)$	$2cT_m + (c+1)T_e + T_p$	$3T_p + (c+2)T_m + (c+3)T_e$

5.2. Experimental Verification

Our experimental environment used Ubuntu-20.04, Parallel Desktop 17 with 8 GB RAM, 64 GB disk space, and a 2 GB CPU. The host system was a macOS Monterey 12.5 running on a MacBook Pro laptop with core Apple M1 and 16 GB RAM. We used the Pair-Based Cryptography (PBC) library [29] for all experiments. In the reproduction experiments [8,15], we split the 1.8 MB dataset into 10,000 equal chunks. When implementing our scheme, we chose to divide the 1.8 MB data set into 1000 data blocks of equal size, with each data block then divided into ten small data blocks of equal size. The test files used are public datasets from the Github "https://ai.stanford.edu/~amaas/data/sentiment/ (accessed on 1 April 2022)" repository. We compared the scheme proposed in this paper with [8,15] in four aspects. It is worth noting that our scheme signed the ten small data blocks uniformly in the experiment. The specific results are described below.

First, we analyze and compare the efficiency of TagGen algorithm. Unlike the other three experiments, we chose to gradually increase the number of data blocks from 1000 to 10,000 in order to count the time required for tag generation. The results are shown in Figure 3. It can see from the figure that the tag generation efficiency of our scheme is much better than the other two schemes. This is because the total number of tags generated by our scheme is about one-tenth that of the other two schemes, making its computational efficiency much better.

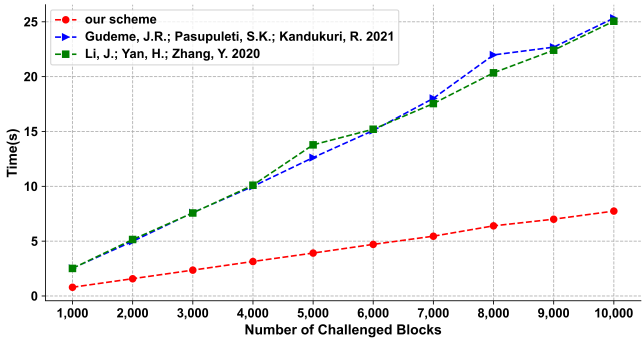

Figure 3. Computation cost of TagGen [8,15].

Because the tags generated by our scheme are much smaller than those of the other two schemes, we allowed our scheme to repeatedly check the same data block during challenge to achieve a fair comparison.

Then, we analyzed the efficiency of Challenge algorithm. In the experiment, we gradually increased the number of challenging data blocks from 1000 to 10,000 and counted the time, as shown in Figure 4. Our scheme takes almost zero time, as does [15]; both are much better than [8]. This is due to the introduction of PRP and PRF, which greatly reduces challenge generation time. Therefore, our scheme has good performance in the challenge generation stage.

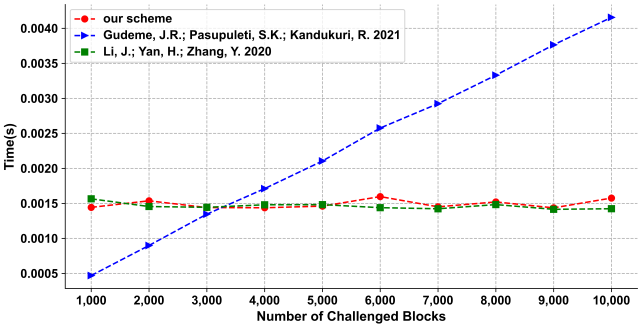

Figure 4. Computation cost of Challenge [8,15].

Next, we compare the performance of the ProofGen algorithm of the three schemes. The results are shown in Figure 5. Obviously, the efficiency of our scheme is low, because in order to make a fair comparison the number of data blocks challenged by our scheme in each challenge is the same as that of the other two schemes, which results in our algorithm spending more computing power to carry out the relevant calculations. It is worth noting, however, that our algorithm checks more data blocks with the same number of challenges.

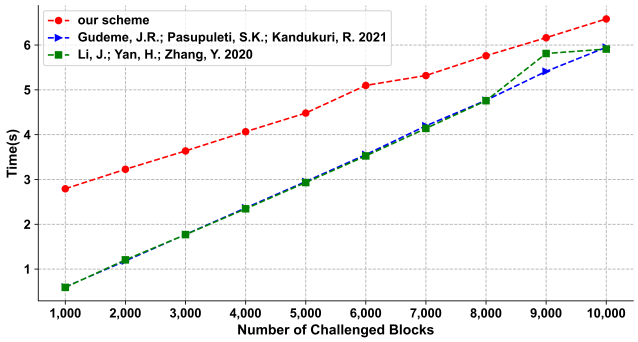

Figure 5. Computation cost of ProofGen [8,15].

The final experiment analyzes the ProofVerify algorithm.The results are shown in Figure 6. For the same reasons as above, this algorithm is less efficient than the other two schemes. However, despite a slight disadvantage in efficiency, the number of data blocks checked is much higher than with the other schemes.

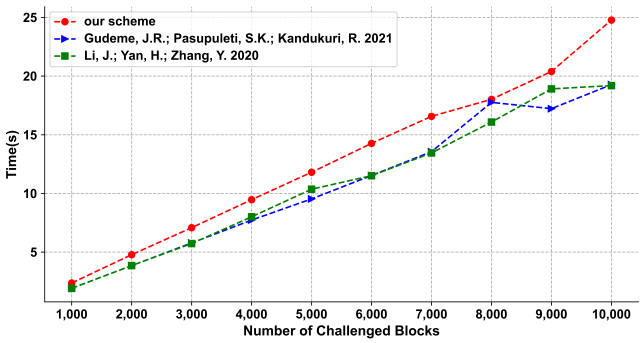

Figure 6. Computation cost of ProofVerify [8,15].

It can be seen from the above experiments that the scheme proposed in this paper is better than both [8,15] in terms of label generation efficiency, while the time cost is slightly higher in proof generation and verification. In addition, the use of PRP and PRF makes the time cost of the proposed scheme in the challenge phase independent of the number of challenged data blocks, which is maintained near a fixed value.

6. Conclusions

In this paper, we propose a certificateless remote data integrity auditing scheme. Our scheme can ensure the privacy of data while providing a data access control algorithm, which makes possible the practical application of the scheme. A detailed security proof guarantees the reasonableness of our scheme. Theoretical analysis and experimental verification show that our scheme has better performance in terms of storage cost and tag generation, making it both efficient and feasible. In actual scenarios, the DO may add, modify, and delete data stored in the cloud. However, this paper does not consider the dynamic updating of data in the design of the audit scheme. In addition, most previous audit schemes consider dynamic updating when signing a single data block. However, the scheme of this paper is to aggregate several data blocks as a whole for signing. Using

the traditional dynamic updating method directly in this paper would lead to substantial waste of computing resources. Therefore, in the future we intend to design an efficient dynamic remote data integrity auditing scheme based on this paper.

Author Contributions: Investigation, G.B. and R.L.; Methodology, G.B.; Software, F.Z.; Supervision, G.B.; Validation, F.Z. and B.S.; Writing—original draft, F.Z.; Writing—review and editing, G.B., F.Z., R.L. and B.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Our study does not require ethical approval, so we choose to exclude this statement.

Informed Consent Statement: Our study does not involve human research, so we choose to exclude this statement.

Acknowledgments: This study was supported by the National Natural Science Foundation of China (No. 61872284) and the Shaanxi Provincial Natural Science Basic Research Project (No. 2021JLM-16).

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

References

1. Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Kissner, L.; Peterson, Z.; Song, D. Provable data possession at untrusted stores. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 28–31 October 2007; pp. 598–609.
2. Juels, A.; Kaliski, B.S., Jr. PORs: Proofs of retrievability for large files. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 28–31 October 2007; pp. 584–597.
3. Garg, N.; Bawa, S.; Kumar, N. An efficient data integrity auditing protocol for cloud computing. *Future Gener. Comput. Syst.* **2020**, *109*, 306–316. [CrossRef]
4. Lu, N.; Zhang, Y.; Shi, W.; Kumari, S.; Choo, K.K.R. A secure and scalable data integrity auditing scheme based on hyperledger fabric. *Comput. Secur.* **2020**, *92*, 101741. [CrossRef]
5. Shen, W.; Qin, J.; Yu, J.; Hao, R.; Hu, J.; Ma, J. Data integrity auditing without private key storage for secure cloud storage. *IEEE Trans. Cloud Comput.* **2019**, *9*, 1408–1421. [CrossRef]
6. Shao, B.; Bian, G.; Wang, Y.; Su, S.; Guo, C. Dynamic data integrity auditing method supporting privacy protection in vehicular cloud environment. *IEEE Access* **2018**, *6*, 43785–43797. [CrossRef]
7. Li, Y.; Zhang, F. An efficient certificate-based data integrity auditing protocol for cloud-assisted WBANs. *IEEE Internet Things J.* **2021**, *9*, 11513–11523. [CrossRef]
8. Gudeme, J.R.; Pasupuleti, S.K.; Kandukuri, R. Certificateless multi-replica public integrity auditing scheme for dynamic shared data in cloud storage. *Comput. Secur.* **2021**, *103*, 102176. [CrossRef]
9. Lu, X.; Pan, Z.; Xian, H. An integrity verification scheme of cloud storage for internet-of-things mobile terminal devices. *Comput. Secur.* **2020**, *92*, 101686. [CrossRef]
10. Li, B.; He, Q.; Chen, F.; Jin, H.; Xiang, Y.; Yang, Y. Auditing cache data integrity in the edge computing environment. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 1210–1223. [CrossRef]
11. Cui, G.; He, Q.; Li, B.; Xia, X.; Chen, F.; Jin, H.; Xiang, Y.; Yang, Y. Efficient verification of edge data integrity in edge computing environment. *IEEE Trans. Serv. Comput.* 2021, *in press*. [CrossRef]
12. Li, B.; He, Q.; Chen, F.; Jin, H.; Xiang, Y.; Yang, Y. Inspecting edge data integrity with aggregated signature in distributed edge computing environment. *IEEE Trans. Cloud Comput.* 2021, *in press*.
13. Zhao, M.; Ding, Y.; Wang, Y.; Wang, H.; Wang, B.; Liu, L. A privacy-preserving tpa-aided remote data integrity auditing scheme in clouds. In Proceedings of the International Conference of Pioneering Computer Scientists, Engineers and Educators, Guilin, China, 20–23 September 2019; pp. 334–345.
14. Yan, H.; Gui, W. Efficient identity-based public integrity auditing of shared data in cloud storage with user privacy preserving. *IEEE Access* **2021**, *9*, 45822–45831. [CrossRef]
15. Li, J.; Yan, H.; Zhang, Y. Identity-based privacy preserving remote data integrity checking for cloud storage. *IEEE Syst. J.* **2020**, *15*, 577–585. [CrossRef]
16. Yang, K.; Jia, X. An efficient and secure dynamic auditing protocol for data storage in cloud computing. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *24*, 1717–1726. [CrossRef]
17. Wang, B.; Li, B.; Li, H.; Li, F. Certificateless public auditing for data integrity in the cloud. In Proceedings of the 2013 IEEE Conference on Communications and Network Security (CNS), Washington, DC, USA, 14–16 October 2013; pp. 136–144.
18. He, K.; Huang, C.; Wang, J.; Zhou, H.; Chen, X.; Lu, Y.; Zhang, L.; Wang, B. An efficient public batch auditing protocol for data security in multi-cloud storage. In Proceedings of the 2013 8th ChinaGrid Annual Conference, Los Alamitos, CA, USA, 22–23 August 2013; pp. 51–56.

19. Yuan, J.; Yu, S. Efficient public integrity checking for cloud data sharing with multi-user modification. In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014; pp. 2121–2129.
20. Jiang, T.; Chen, X.; Ma, J. Public integrity auditing for shared dynamic cloud data with group user revocation. *IEEE Trans. Comput.* **2015**, *65*, 2363–2373. [CrossRef]
21. Zhang, Y.; Xu, C.; Liang, X.; Li, H.; Mu, Y.; Zhang, X. Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation. *IEEE Trans. Inf. Forensics Secur.* **2016**, *12*, 676–688. [CrossRef]
22. Li, Y.; Yu, Y.; Min, G.; Susilo, W.; Ni, J.; Choo, K.K.R. Fuzzy identity-based data integrity auditing for reliable cloud storage systems. *IEEE Trans. Dependable Secur. Comput.* **2017**, *16*, 72–83. [CrossRef]
23. Shen, W.; Qin, J.; Yu, J.; Hao, R.; Hu, J. Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage. *IEEE Trans. Inf. Forensics Secur.* **2018**, *14*, 331–346. [CrossRef]
24. Zhao, Q.; Chen, S.; Liu, Z.; Baker, T.; Zhang, Y. Blockchain-based privacy-preserving remote data integrity checking scheme for IoT information systems. *Inf. Process. Manag.* **2020**, *57*, 102355. [CrossRef]
25. Lan, C.; Li, H.; Wang, C. Cryptanalysis of "Certificateless remote data integrity checking using lattices in cloud storage". In Proceedings of the 2020 10th International Conference on Information Science and Technology (ICIST), Lecce, Italy, 4–5 June 2020; pp. 134–138.
26. Tian, J.; Wang, H.; Wang, M. Data integrity auditing for secure cloud storage using user behavior prediction. *Comput. Secur.* **2021**, *105*, 102245. [CrossRef]
27. Yang, C.; Liu, Y.; Zhao, F.; Zhang, S. Provable data deletion from efficient data integrity auditing and insertion in cloud storage. *Comput. Stand. Interfaces* **2022**, *82*, 103629. [CrossRef]
28. Dhelim, S.; Aung, N.; Kechadi, T.; Ning, H.; Chen, L.; Lakas, A. Trust2Vec: Large-Scale IoT Trust Management System based on Signed Network Embeddings. *IEEE Internet Things J.* 2022, *in press*.
29. Lynn, B.; Shacham, H.; Steiner, M.; Cooley, J.; Figueiredo, R.; Khazan, R.; Kosolapov, D.; Bethencourt, J.; Miller, P.; Cheng, M.; et al. PBC Library. 2006; Volume 59, pp. 76–99. Available online: http://crypto.stanford.edu/pbc (accessed on 20 May 2020).