electronics

Article

Certificateless Data Integrity Auditing in Cloud Storage with a Designated Verifier and User Privacy Preservation

Genqing Bian [1], Xusen Guo [1,*], Rong Li [2], Wenjing Qu [1] and Yu Zhao [3]

[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
[3] School of Management, Xi'an University of Architecture and Technology, Xi'an 710311, China
* Correspondence: guoxusen@xauat.edu.cn; Tel.: +86-18821673397

Abstract: With the rapid development of science and technology, enterprises will provide their customers with cloud data storage services. These massive amounts of data bring huge management costs to enterprises. Therefore, enterprises choose to store their data in professional cloud service providers and have third-party auditors check the integrity of cloud data to ensure security. Although the appearance of auditors reduces the enormous calculation pressure on enterprises, if the number of auditors is not limited, it will also bring an expensive management burden to enterprises. At the same time, in the process of performing data integrity auditing on behalf of the enterprise, auditors may be interested in some sensitive information of the enterprise's customers (such as customer's identity and specific content of customer data). Therefore, this paper proposes a remote data integrity auditing scheme based on designated verifiers. An essential feature of the scheme is that the auditor cannot obtain any customer's identity information and data in the process of auditing; data integrity, the anonymity of the user's identity, and data privacy are maintained in the process of auditing. Both theoretical analysis and experimental results show that our scheme is efficient and feasible.

Keywords: cloud storage; data integrity auditing; privacy preserving; designated verifier

Citation: Bian, G.; Guo, X.; Li, R.; Qu, W.; Zhao, Y. Certificateless Data Integrity Auditing in Cloud Storage with a Designated Verifier and User Privacy Preservation. *Electronics* **2022**, *11*, 3901. https://doi.org/10.3390/electronics11233901

Academic Editor: Antonio Brogi

Received: 13 October 2022
Accepted: 19 November 2022
Published: 25 November 2022

1. Introduction

With the advent of the era of big data, human production activities will produce massive amouts of data every day. The storage of these data will bring expensive maintenance costs and economic expenses to the data owner (DO). To address this problem, several companies are launching plans to provide cloud storage services for individuals or companies at little or no cost. From DO's point of view, this service will undoubtedly be a huge convenience, freeing them from the hectic maintenance of data. However, this service also brings a new problem, as the DO loses direct control over important data. In this case, the cloud service provider (CSP) may delete data that the DO does not often access or never accesses in order to save costs. Alternatively, the hardware servers used by the CSP to store original data may fail, resulting in data loss. In either case, this can cause severe financial losses to the DO.

Given the above background, it is natural for the DO to be concerned about the integrity of the data stored in the cloud and to be eager to have a mechanism to help them check the integrity of the outsourced data. Therefore, various remote data integrity checking (RDIC) mechanisms, such as provable data possession (PDP), have been used extensively in the past decades [1–10]. The PDP model is divided into private and public auditing depending on the types of the verifier. Both methods enable data integrity auditing. In private auditing, the DO communicates with the CSP to check the integrity of the data stored in the cloud, which undoubtedly imposes a substantial computational overhead on the DO. In public auditing, anyone can check the integrity of the data, so as to reduce

the communication burden on the DO, and this operation is usually delegated to a third-party auditor (TPA) [11,12]. Here the TPA can be an individual or an authority with more computing power than the DO.

Since the amount of data that the DO uploads to the cloud is tremendous, and data integrity auditing needs to be checked consistently over time, most RDIC mechanisms reduce the workload by checking the integrity of only a part of the data each challenge rather than all the data stored in the cloud. In the PDP model, the TPA randomly selects some data blocks stored in the cloud and sends the challenge information to the CSP. When the CSP receives the message, it selects the challenged data blocks and the corresponding signatures to generate a proof to send to the TPA in a response to this challenge. After receiving proof from the CSP, TPA verifies the proof to determine whether the CSP has stored DO data well [13]. In the cloud auditing model, TPA is generally considered to be honest-but-curious; thus, TPA will honestly judge whether user data are complete but will be curious about the data. Therefore, although TPA can be a great convenience for the DO, there is a risk that the original data will be leaked to the TPA during auditing process. In addition, many existing PDP schemes have complex certificate management issues, and the DO must properly store private keys that have been certified by the public key infrastructure (PKI) [12,14,15]. To reduce the storage burden of the DO, Shen et al. designed a signature scheme that using the DO's biological (such as fingerprint) instead of traditional private keys [16]. To avoid complex certificate management issues, it is more common to use the identity-based PDP (ID-PDP) signature scheme [17–20].

These large numbers of RDIC protocols have often adopted by enterprises since they were proposed. These auditing protocols have been widely used in healthcare, finance, transportation and other fields. Its application in the field of transportation has attracted much attention from the market. In today's growing number of private cars, the dashcam has become a must-have device for every car. For a single car owner, uploading the data recorded by the camera to the cloud and appointing a verifier to check it is a very troublesome task. Fortunately, in order to give better feedback to their customers, enterprises will take the initiative to help them store data in the cloud and check the integrity of the data. This service can reduce the local storage burden on the DO, and the data are also an important reference for enterprises to improve their services. The enterprise stores user data in the CSP and downloads relevant data from the CSP when required by itself or users. In this process, how to ensure the privacy of DO information is a problem worth considering. In the field of transportation, the identity of the DO needs to be properly protected, as well as his own data. Identity information can sometimes reveal a lot of valuable information. If there is an auditing scheme that can ensure the anonymity of the user's identity and the privacy of the data, TPA can only judge whether the data being checked is complete when checking the integrity of the data and cannot obtain any other useful information. In [21], Yan et al. designed an efficient ID-PDP scheme to ensure the anonymity of the DO while implementing data integrity auditing; TPA can only judge whether the data are complete during the auditing process without obtaining any information about the identity of the DO. However, this article does not consider data privacy security. In addition, while the advent of TPA eased the computational burden on the DO, it was prone to administrative chaos because anyone could audit the data. Therefore, some entity wants to restrict the verifier who checks the integrity of data to obtain the identity; that is, they can only hope that the designated verifier (DV) can complete the work [22]. To address this problem, we proposed a certificateless PDP scheme for auditing the customer's data stored in the cloud through the company's designated verifier. In this paper, we envisage the application of data integrity auditing in the field of the Internet of Vehicles. By implementing the scheme in this paper, automobile companies can provide better services to their customers on the premise of ensuring safety.

1.1. Our Contributions

To summarize, both ensuring the anonymity of identity information and setting up a designated verifier to check the integrity of data are urgent problems to be solved. However, maintaining the anonymity of identity and setting up a designated verifier seem to be logically contradictory things. If the DO wants to set a designated verifier, the verifier must know who appointed it, so the anonymity of the DO's identity cannot be guaranteed. In order to solve this problem, we introduce a new entity, CP, in the scheme. In our envisioned scenario, CP can be the general agent of a car company. The CP serves many DOs, setting designated verifiers for DOs to help them check data. From the point of view of the DV, when implementing the audit scheme designed in this thesis, it only knows that it is designated by CP but does not know which DO's data it is checking. Through such a setting, this paper not only guarantees the anonymity of the DO's identity but also sets up a designated verifier. The main contributions of this paper are as follows:

1. We have designed a PDP scheme that uses certificateless signature technology, in which the data auditing work is performed by the designated verifier. In the process of data auditing, the DV cannot obtain any other information except the result of whether the data are integrated. Specifically, the auditing process ensures both the privacy of data and the anonymity of the DO. In the scheme we designed, TPA can only judge whether the data checked are complete when performing the audit task and cannot obtain any relevant information about the data and the identity information that the data have; that is, TPA only knows that it has checked the data of a certain user of a company and does not know the specific identity of this user.
2. Most PDP schemes are in one-to-one mode for the digital signature of data blocks, which greatly burdens communication and storage. To solve this problem, the scheme we designed splits the raw data into a matrix before digital signature, with each row containing ten blocks of raw data. In the digital signature stage, each row of the matrix is signed as a whole, so that the number of tag blocks is reduced to one-tenth of the number of data blocks, which greatly reduces the burden of storage and communication.
3. We give the provable security analysis for our scheme in a random oracle model. Moreover, we compare the proposed scheme with other schemes, and the results show that our scheme has a good performance in efficiency.

1.2. Related Work

In 2007, Ateniese et al. [23] proposed for the first time to use the PDP model to check the integrity of data stored in the cloud. The homomorphic verification technique used in this paper can aggregate multiple proofs in the auditing process into a single value, which greatly reduces the communication burden. In the same year, Juels et al. [24] proposed the PoR model, which realized data integrity auditing but did not support public verification.

In 2010, Wang et al. [14] expanded the PDP model and proposed a data integrity auditing scheme supporting public auditing and a dynamic update on the basis of Merkle hash trees (MHT). However, this article is based on public audits, which is extremely unfriendly to the anonymity of users' identities. In 2016, Yan et al. [25] realized data integrity auditing and dynamic update operations through an operation record table (ORT). However, their method brought the problem of increased computational overhead. Li et al. [22] made specific improvements to the traditional PDP model, so that the work of checking data integrity can only be carried out by the verifier designated by the user. However, this paper does not consider the dynamic updating of data, and there are some flaws in the security proof. Sun et al. [26] designed a new data authentication structure, P-ATHAT, by introducing trapdoor and BLS signature to MHT. However, the paper was ill-considered in terms of privacy. In order to minimize the computational complexity of the DO, Garg et al. [15] proposed a PDP scheme and proved the security of the scheme through CDH assumptions. As a protocol implemented on the mobile side, the solution is not lightweight enough. In order to better reduce the computational burden on the

DO, Lu et al. [12] designed an MHT-based PDP scheme, in which the tags generated by the DO are changed to those generated by TPA. However, the scheme gives TPA too many rights, so there is a risk of man-in-the-middle attacks. Considering that most of the traditional PDP scheme uses signature technologies, such as RSA or BLS, which will bring great computational overhead and low efficiency, Zhu et al. [27] designed a PDP scheme with privacy protection and higher efficiency based on ZSS signature technology. However, the above papers involve a complex certificate management process. In 2020, Ning et al. [28] implemented an auditing scheme with the help of the Hyperledger fabric, so that each audit task can dynamically select the TPA.

Identity-based signature technology can effectively solve the above problems. Shang et al. [29] used the identity-based PDP model to achieve dynamic data updating and auditing operations. However, the disadvantages of privacy and large computational overhead make this scheme not suitable for most scenarios. Zhao et al. [30] designed a big data dynamic auditing method based on fuzzy identity by combining MHT and index logic tables (ILT), which simplified the interaction process between DO and CSP in dynamic updates. Again, the paper does not do a good job of privacy protection. Considering the privacy of important information, Shen et al. [31] designed an identity-based PDP model so that user information would not be leaked to any malicious entity in the auditing process. For the first time, Peng et al. [17] proposed an identity-based PDP scheme with full dynamic updates and multi-replica batch checking. Unfortunately, the calculation overhead of this scheme is still a bit high. Li et al. [18] proposed identity-based authentication technology, which ensures the privacy of data in the process of data auditing. However, the security proof of this scheme is not quite correct. Yang et al. [19] designed a compressive secure cloud storage scheme inspired by the Goldreich–Goldwasser–Halevi (GGH) cryptosystem. However, the introduction of this mechanism increases the actual overhead of the scheme. Tian et al. [20] designed a scheme that supports user behavior prediction, which saves resources by setting a safe time to avoid repeated verification of the same data block in a short period of time. The problem with identity-based signature technology is that key management is too centralized, and there is a risk of being attacked. In 2022, Li et al. [32] proposed a lightweight auditing scheme based on certificates and provided a security model.

The use of certificateless signature technology can solve the above problems. Wang et al. [33] designed a lightweight PDP scheme based on asymmetric bilinear mapping. In this scheme, the certificate participates in the decryption operation as a part of the key. In the same year, Hong et al. [34] proposed an efficient certificateless auditing scheme based on the Internet of Things. However, the data in the industrial field are constantly flowing, and this scheme does not consider the dynamic updates of data security. Considering the needs of group users, Li et al. [35] designed a certificateless PDP scheme. In order to improve the availability and durability of data, Zhou et al. [36] proposed a multi-copy dynamic auditing scheme with certificateless signatures, which not only guarantees the privacy of data, but also supports the dynamic update operation of multi-copy data. In the same year, Jaya et al. [37] designed a multi-copy audit protocol, which has excellent performance in computational efficiency. Unfortunately, the above studies are not perfect in terms of identity anonymity.

1.3. Organization

The remainder of this paper is organized as follows. In Section 2, we introduce the basic knowledge and give the security model of our scheme. In Section 3, we give the detailed structure of the proposed scheme. In Section 4, a provable security analysis is given. In Section 5, we evaluated our scheme both theoretically and experimentally. Finally, the conclusion of this paper is presented in Section 6 [4].

2. Preliminaries

In this part, we first introduce the system model and basic knowledge involved in this paper, then give the outline of our scheme and the security model. Table 1 below shows some of the mathematical symbols involved in this paper and their corresponding meanings.

Table 1. Mathematical notations.

Notation	Description
G_1, G_2	Two multiplicative cyclic groups
p	A large prime number
g	A generator of the multiplicative group G_1
e	Bilinear map
H_1, H_2	Two secure cryptographic hash functions
$u_1, u_2, \ldots, u_s$	s 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, company (CP) and DV. The relationship between them is shown in Figure 1.

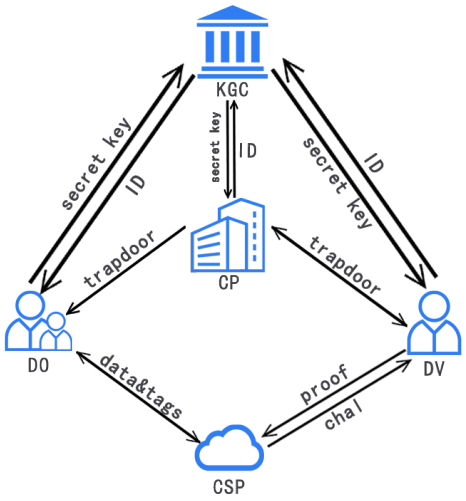

Figure 1. System model.

CP: The CP is responsible for assigning verifiers to its clients to check the integrity of data stored in the cloud.

DO: The DO cuts and chunks the local data while generating tags and then sends them to the CSP.

CSP: The CSP has abundant storage space and powerful computing ability. It can provide data storage services for the DO and proof of data integrity for the DV when receiving a data integrity challenge.

KGC: The KGC is an organization that is responsible for distributing keys. Each time a request is received, the KGC generates part of the private key for the client.

DV: The DV is designated by the CP, which checks whether the data are completely stored in the cloud. The DV has enough computing power to complete data integrity verification.

2.2. Bilinear Maps

Let G_1 and G_2 be two multiplicative cyclic groups with the same large prime order p. g is a generator of G_1. e is a bilinear map that satisfies $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

Bilinearity: for $\forall m, n \in G_1$ and $\forall a, b \in Z_p^*$, there are $e\left(m^a, n^b\right) = e(m, n)^{ab}$.

Computability: for $\forall m, n \in G_1$, there is an efficiently algorithm to compute $e(m, n)$.

Non-degeneracy: $\exists m, n \in G_1$, there is $e(m, n) \neq 1_{G_2}$.

2.3. Security Assumptions

CDH (Computational Diffie–Hellman) Problem: Let G_1 be a multiplicative cyclic group. g is a generator of G_1. Given the tuple $\left(g, g^a, g^b\right)$, where $a, b \in Z_p^*$ is unknown, the CDH problem calculates g^{ab}.

CDH Assumption: For any probabilistic polynomial time (PPT) adversary Λ, the advantage for Λ to solve the CDH problem in G_1 is negligible. Assume ε is a negligible value; it can be defined as:

$$Adv_{G_1\Lambda}^{CDH} = \Pr\left[\Lambda\left(g, g^a, g^b\right) = g^{ab} : a, b \overset{R}{\leftarrow} Z_p^*\right] \leq \varepsilon \tag{1}$$

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

DL Assumption: For any PPT adversary Λ, the advantage for Λ to solve the DL problem in G_1 is negligible. Assume ε is a negligible value; it can be defined as:

$$Adv_{G_1\Lambda}^{DL} = \Pr\left[\Lambda(g, g^x) = x : x \overset{R}{\leftarrow} Z_p^*\right] \leq \varepsilon \tag{2}$$

2.4. Outline of Our Scheme

Our scheme contains eight algorithms, and their functions are described as follows:

$Setup(1^k) \rightarrow (params, msk)$: This algorithm inputs security parameter k, outputs public parameters $params$ and system secret key msk.

$Extract(params, ID, msk) \rightarrow pp_{ID}$: This algorithm is run by the KGC, which is used to generate part of the private key. After receiving the customer ID, the algorithm uses the master key msk to generate part of the security key for the customer.

$KeyGen(params, pp_{ID}) \rightarrow (sk, pk)$: This algorithm is run by a customer to generate his own key pair.

$TrapdoorGen(params, pk, sk) \rightarrow \alpha$: This algorithm generates a trapdoor through the key pair of the CSP and DV.

$TagGen(params, sk, F, \alpha) \rightarrow (\Phi, ftag)$: This algorithm is run by the DO, which is used to generate tags for data blocks and the file name.

$Challenge(params, c) \rightarrow chal$: This algorithm is run by DV to generate a data integrity challenge request $chal$ for the file named Fid.

$ProofGen(params, F, \Phi, chal) \rightarrow P$: After receiving $chal$, the algorithm generates the corresponding proof P with the data blocks and tags.

$ProofVerify(params, P, ftag, chal, \alpha) \rightarrow \{0, 1\}$: The validity of P is verified by this algorithm to determine whether the data are intact. If the output result is 1, it proves that the data are still saved safely; otherwise, it has been damaged.

To further illustrate our proposed scheme, the following Figure 2 is a flowchart of the algorithm, where n is the total number of tags to generate hypothetically, and k is the number of challenges.

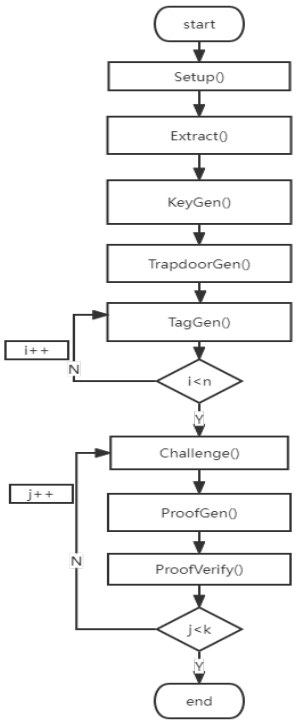

Figure 2. Algorithm flowchart.

2.5. Security Model

To elaborate on the security of the scheme proposed in this paper, we design a series of games between a challenger β and an adversary A. Although the scheme designed in this paper involves multiple users, in order to simplify the proof process, we used a single user with an identity ID as an example in the security model. It is worth noting that the challenger β represents the DO, and the adversary A represents a malicious cloud. The most basic game rules are as follows:

Setup: The challenger β executes system initialization to obtain the public parameters, params, and the master key *msk*. Then the challenger β sends *params* to the adversary A and keeps *msk* secret.

Queries: For the game to proceed effectively, the following interactions can be performed between the challenger β and the adversary A.

Hash queries: The adversary A can send a series of different *Hash queries* to the challenger β. When the challenger β receives relevant query information, it will use the resources at its disposal to perform relevant calculations and feedback the results to the adversary A.

Partial secret public key query: In order to forge legal proof, the adversary A can query the public and private keys of the DO whose identity is ID, and the challenger β sends the result to the adversary A when it receives the relevant query information.

Tag query: The adversary A can randomly select a data block m_i and query its corresponding tag. The challenger β executes *TagGen* to generate the tag and sends it to the adversary A.

Integrity proof queries: The adversary A can also query the integrity proof of any set of data blocks. When receiving a query request, challenger β performs relevant calculations and shares the results with adversary A.

Challenge: At this stage, the challenger β generates a challenge set of data blocks and sends it to the adversary A. The adversary generates a proof and sends it to the challenger β after receiving the challenge information.

Forge: After receiving the challenge information, the adversary A generates a data integrity proof P and sends it to the challenger β. If the proof can be verified by β, it is considered that the adversary A has won this game.

In the above model, the goal of the adversary is to pass the verification of the challenger β by using a forged proof P^* for the challenged blocks. Obviously, we need to prove that the adversary A cannot generate a valid proof without fully grasping the data blocks involved in the challenge information.

Definition 1. *If the probability of A winning the game is non-negligible, there is a knowledge extractor that can listen to the communication between the adversary A and the challenger β.*

Our scheme focuses on protecting users' information from being disclosed to unauthorized organizations. Privacy information in our scheme includes both original data and the user's identity. The DV tries to obtain original data and distinguish the identity of the DO during the data integrity auditing process. Thus, the scheme should not only ensure data privacy against DV but also enable the DO's anonymity against DV.

Definition 2. *A certificateless data integrity auditing scheme for important information is privacy preserving if the DV cannot distinguish the identity of DO and cannot obtain the DO's original data during the data integrity auditing process.*

3. The Proposed Scheme

In this section, we elaborate on the certificateless data integrity auditing scheme with privacy preservation and a designated verifier.

The DO divides the local data file F into n blocks with a fixed length, and each data block contains s small data blocks with the same length. That means

$$\begin{aligned} F &= \{m_1, \ldots, m_n\} \\ &= \{(m_{11}, m_{12}, \ldots, m_{1s}), \ldots, (m_{n1}, m_{n2}, \ldots, m_{ns})\}. \end{aligned} \tag{3}$$

$Fid \in \{0,1\}^*$ is a unique symbol for the file F. The details of the algorithms involved in our scheme are shown below.

$Setup(1^k) \rightarrow (params, msk)$: Given a security parameter k, KGC randomly selects a large prime p with the feature $|p| = k$. Then, KGC selects two multiplicative cyclic groups G_1 and G_2 with the same order p, respectively. g is a generator of G_1, and $(u_1, u_2, \ldots, u_s) \in (G_1)^s$ are all random elements of the group G_1. H_1, H_2 are two cryptographic hash functions, which satisfy that $H_1 : \{0,1\}^* \longrightarrow G_1, H_2 : \{0,1\}^* \longrightarrow G_1$. e is a bilinear map acting on G_1, $G_2 : G_1 \times G_1 \longrightarrow G_2$. $\pi : Z_p^* \times \{1,2,\ldots,n\} \longrightarrow \{1,2,\ldots,n\}$ is a pseudo-random permutation (PRP), and $\varphi : Z_p^* \times Z_p^* \longrightarrow Z_p^*$ is a pseudo-random function (PRF). Then, KGC generates the master secret key $msk = x$, where x is randomly selected from Z_p^*. The master public key mpk is calculated by $mpk = g^x$. After doing the above work, KGC will publish $params = (G_1, G_2, g, p, u_1, u_2, \ldots, u_s, e, mpk, H_1, H_2, \pi, \varphi)$ but keep msk private.

$Extract(params, ID, msk) \rightarrow pp_{ID}$: When the ID representing the customer's identity is received, KGC calculates the partial private key of this customer by the equation $pp_{ID} = H_1(ID)^x$ and then sends it to customer through a secure channel. By this method, DO, DV and CP obtain their corresponding partial private keys pp_{ID_O}, pp_{ID_V}, pp_{ID_C}.

$KeyGen(params, pp_{ID}) \rightarrow (sk, pk)$: Take the DO as an example. After receiving the pp_{ID_O} sent by KGC, the DO first checks the equation $e(pp_{ID_O}, g) = e(H_1(ID_O), mpk)$. If the validation fails, the DO terminates this algorithm and applies to KGC for a partial private key again. Otherwise, the DO selects a secret value $s_O \in Z_p^*$ and calculates $pk_O = g^{s_O}$. Thus, the DO combines s_O with pp_{ID_O} as his private key $sk_O = (s_O, pp_{ID_O})$ and published public key pk_O. By this method, DV and CP obtain their corresponding key pair $(sk_V = (s_V, pp_{ID_V}), pk_V = g^{s_V})$, $(sk_C = (s_C, pp_{ID_C}), pk_C = g^{s_C})$, respectively.

$TrapdoorGen(params, pk, sk) \rightarrow \alpha$: The CP uses its own private key and the public key of the DV to compute trapdoor $\alpha = pk_V^{s_{ID_C}}$ and sends it to the DO. Obviously, the DV can obtain the same result by calculating $\alpha = pk_C^{s_{ID_V}}$.

$TagGen(params, sk_O, F, \alpha) \rightarrow (\Phi, ftag)$: The DO generates a tag $ftag$ for the file F by using a certificateless signature technique such as [33]. The CSP can verify the validity of $ftag$ through the same signature technique. Then, the DO calculates the tag by computing Equation (4):

$$\sigma_i = \left(pp_{ID_O} \cdot H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}} \right)^{1/s_O} \tag{4}$$

for each $i \in \{1, 2, \ldots, n\}$. Denote the collection of tags as $\Phi = \{\sigma_i | i \in \{1, 2, \ldots, n\}\}$. $1/s_O$ is the multiplicative inverse element of s_O in Z_p^*. Finally, the DO transfers message $\{F, \Phi, ftag\}$ to the CSP. A single tag can be verified by $e(\sigma_i, pk_O) = e(H_1(ID_O), mpk) \cdot e(H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}, g)$.

$Challenge(params, c) \rightarrow chal$: The DV can perform multiple integrity challenges and randomly select some data blocks for each challenge. Suppose one of the challenges check c blocks and the DV randomly selects two values, $k_1, k_2 \in Z_p^*$. Then, the DV sends the challenge request $chal = (c, k_1, k_2)$ to the CSP.

$ProofGen(params, F, \Phi, chal) \rightarrow P$: After receiving $chal$ from the DV, the CSP randomly selects two elements $r \in G_1$ and $k \in Z_p^*$. The parameter set $C = \{(\alpha_l, \beta_l) | l \in \{1, 2, \ldots, c\}\}$ is calculated, where $\alpha_l = \varphi(k_1, l), \beta_l = \pi(k_2, l)$. Then, the CSP calculates

$$\begin{cases} P_1 = r \cdot H_1(ID_O)^{\sum_{i=1}^{c} \alpha_i} \\ P_2 = e(r, mpk) \cdot e(\prod_{i=1}^{c} \sigma_{\beta_i}^{\alpha_i}, pk_O) \\ M_j = \sum_{i=1}^{c} \alpha_i m_{\beta_i j} + k, j \in \{1, 2, \ldots, s\} \\ R = \prod_{j=1}^{s} u_j^{-k} \end{cases} \tag{5}$$

Here, $-k$ is the additive inverse of k in Z_p^*. Finally, the CSP sends $P = \{P_1, P_2, M_1, M_2, \ldots, M_s, R\}$ to DV.

$ProofVerify(params, P, Fid, chal, \alpha) \rightarrow \{0, 1\}$: After receiving the proof P from CSP, the DV calculates $C = \{(\alpha_l, \beta_l) | l \in \{1, 2, \ldots, c\}\}$, where $\alpha_l = \varphi(k_1, l), \beta_l = \pi(k_2, l)$. Then, it verifies Equation (6):

$$P_2 = e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j} \cdot R, g) \tag{6}$$

If Equation (6) holds, the data blocks are well preserved in this challenging period, and it returns 1. Otherwise, it means that the data has been damaged, and it returns 0.

4. Security Proof

In this section, we give the security proof of our scheme. The security of this paper mainly involves five aspects: correctness, soundness, privacy preservation, trapdoor security, and detectability. For each aspect, this section gives the following detailed proof.

4.1. Correctness Proof

The scheme designed in this paper contains the three core functions of key generation, tag verification, and proof verification, so a detailed proof of its correctness is required.

First, we prove the correctness of the partial private key generated by KGC. When KGC outputs a correct partial private key for the client, it obviously has the following formula:

$$e(pp_{ID}, g) = e(H_1(ID)^x, g) = e(H_1(ID), mpk) \tag{7}$$

Therefore, the correctness of the generated part of the private key has been proven.

Then, we prove the correctness of the tag verification. In this paper, Equation (4) shows how to verify the correctness of a signal tag. With the help of the nature of the bilinear map, the correctness of Equation (4) can be proved as follows:

$$\begin{aligned} e(\sigma_i, pk) &= e(\left(pp_{ID_O} \cdot H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}} \right)^{1/s_O}, g^{s_O}) \\ &= e(pp_{ID_O} \cdot H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}, g) \\ &= e(H_1(ID_O)^x \cdot H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}, g) \\ &= e((H_1(ID_O)^x, g) \cdot e(H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}, g) \\ &= e(H_1(ID_O), mpk) \cdot e(H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}, g) \end{aligned} \tag{8}$$

Equation (6) explains how the DV checks the proof of KGC feedback, and the proof of its correctness is as follows.

$$\begin{aligned} P_2 &= e(r, mpk) \cdot e(\prod_{i=1}^{c} \sigma_{\beta_i}^{\alpha_i}, pk_O) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c}((pp_{ID_O} \cdot H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{1/s_O})^{\alpha_i}, g^{s_O}) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c}((pp_{ID_O} \cdot H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{\alpha_i})^{1/s_O}, g^{s_O}) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c}(pp_{ID_O} \cdot H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{\alpha_i}, g) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c} pp_{ID_O}^{\alpha_i}, g) \cdot e(\prod_{i=1}^{c}(H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{\alpha_i}, g) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c}(H_1(ID_O)^x)^{\alpha_i}, g) \cdot e(\prod_{i=1}^{c}(H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{\alpha_i}, g) \\ &= e(r, g^x) \cdot e(\prod_{i=1}^{c}(H_1(ID_O)^{\alpha_i})^x, g) \cdot e(\prod_{i=1}^{c}(H_2(Fid||\beta_i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{\beta_i j}})^{\alpha_i}, g) \\ &= e(r \cdot \prod_{i=1}^{c} H_1(ID_O)^{\alpha_i}, g^x) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{i=1}^{c}\prod_{j=1}^{s} u_j^{\alpha_i m_{\beta_i j}}, g) \\ &= e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{\sum_{i=1}^{c} \alpha_i m_{\beta_i j} + k - k}, g) \\ &= e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{\sum_{i=1}^{c} \alpha_i m_{\beta_i j} + k} \cdot \prod_{j=1}^{s} u_j^{-k}, g) \\ &= e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j} \cdot R, g) \end{aligned} \tag{9}$$

Thus, the equality reasoning for the proof of the verification algorithm has been completed.

4.2. Soundness Proof

Theorem 1 (Unforgeability)**.** *If the probability of the adversary A winning the game is negligible, then our scheme satisfies the proof unforgeability under Definition 1.*

Proof. We will prove through several games that if the adversary A successfully forges a proof P that can be verified by the challenger β in the absence of complete data, there is a knowledge extractor that can solve the CDH question or DL question by intercepting the communication information between A and β. □

Game 0: *Game* 0 is already defined in the security model in Section 2; we therefore will not go into too much detail here.

Game 1: *Game* 1 and *Game* 0 have the same rules except for one detail. In addition to challenging adversary A, the challenger β also maintains a local list of information about each challenge. The challenger β carefully checks the proof information P returned by each adversary A. If the proof information returned by the adversary A passes the verification algorithm, and the P_2 in it is not equal to the expected P_2, which means that there is at least one tag that differs from the true value, the challenger β terminates the game and declares failure.

Analysis: If the adversary A wins *Game* 1 with non-negligible probability, then there exists a knowledge extractor that can solve the CDH problem with non-negligible probability. In this process, the knowledge extractor takes the place of the challenger β to talk to the adversary A. Given $g, g^{\alpha}, h \in G_1$, the knowledge extractor's goal is to compute h^{α}.

If the adversary A wins the game, it means that it has successfully forged a verifiable proof $P' = \{P_1, P_2', M_1', M_2', \ldots, M_s', R\}$. Then, the knowledge extractor has:

$$P_2' = e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j'} \cdot R, g) \tag{10}$$

Assuming the correct proof is $P = \{P_1, P_2, M_1, M_2, \ldots, M_s, R\}$, then it has:

$$P_2 = e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j} \cdot R, g) \tag{11}$$

Obviously, there is at least one difference between $M_1, M_2, \ldots, M_s$ and $M_1', M_2', \ldots, M_s'$, otherwise $P_2 = P_2'$. The knowledge extractor randomly selects $r_i \in Z_q^*$ for each i $(1 \le i \le c)$ in the challenge, then sets $u_j = g^a \cdot h^b$ where $a, b \in Z_q^*$ and $1 \le j \le t$. Thus, the knowledge extractor has $\prod_{j=1}^{t} u_j^{m_{ij}} = \prod_{j=1}^{t} [g^a \cdot h^b]^{m_{ij}} = (g^a)^{\sum_{j=1}^{t} m_{ij}} \cdot (h^b)^{\sum_{j=1}^{t} m_{ij}}$. After doing that, the extractor randomly selects an element $x \in Z_q^*$ as msk and then performs *Extract* and *KeyGen* to generate the private key $sk = (sk_1, sk_2) = (H_1(ID)^x, s)$. Then the extractor randomly selects an element $k \in Z_q^*$ and sets $pk = g^s = (g^{\alpha})^k$, which means $s = k \cdot \alpha$. Finally, for each i, the knowledge extractor computes:

$$H_2(ID||fname||i) = g^{r_i} / (g^a)^{\sum_{j=1}^{t} m_{ij}} \cdot (h^b)^{\sum_{j=1}^{t} m_{ij}} \tag{12}$$

Thus, the tag can be comouted by $\sigma_i = \left(pp_{ID} \cdot H_2(Fid||i||\alpha) \cdot \prod_{j=1}^{s} u_j^{m_{ij}}\right)^{1/s} = H_1(ID)^x \cdot (g^{r_i})^{1/s}$. Dividing the two verification equations, the knowledge extractor has

$$e(\sigma^* / \sigma, pk) = e(\prod_{j=1}^{s} u_j^{(M_j^* - M_j)}, g) \tag{13}$$

Simplifying this equation even further, the knowledge extractor has:

$$\begin{aligned} e(\sigma^* \cdot \sigma^{-1}, pk) &= e((\prod_{j=1}^{s} u_j^{(M_j^* - M_j)}, g) \\ &= e((g^a)^{\sum_{j=1}^{s}(M_j^* - M_j)} \cdot (h^b)^{\sum_{j=1}^{s}(M_j^* - M_j)}, pk^{-k\alpha}) \\ &= e((g^a)^{-k \cdot \sum_{j=1}^{s}(M_j^* - M_j)} \cdot (h^b)^{-k \cdot \sum_{j=1}^{s}(M_j^* - M_j)}, pk^{\alpha}) \\ &= e((h^b g^a)^{-k \cdot \sum_{j=1}^{s}(M_j^* - M_j)}, pk^{\alpha}) \end{aligned} \tag{14}$$

Finally, the knowledge extractor can solve the CDH problem with the following equation:

$$h^{\alpha} = (\sigma^* \cdot \sigma^{-1} \cdot (g^{\alpha})^{-a \cdot k \cdot \sum_{j=1}^{t} \cdot \triangle M_j})^{1/(b \cdot k \cdot \sum_{j=1}^{t} \cdot \triangle M_j)} \tag{15}$$

It can be easy to find that the probability for this equation to fail is a negligible value $1/p$. Then, the knowledge extractor can find a solution to the CDH problem with a high probability $1 - 1/p$. This contradicts with the CDH assumption.

Game 2: This game is run between the challenger β and the adversary A in the same manner as *Game* 1 with one difference. The challenger β still observes each instance of the challenge and response proof, while the challenger β declares failure and aborts this game if there exists M^* not equal to the correct M.

Analysis: Given a DL instance $g, h \in G_1$, it can construct an extractor whose purpose is to calculate a value $\alpha \in Z_p^*$ that satisfies $h = g^{\alpha}$. Suppose there is a correct proof $P = \{P_1, P_2, M, R\}$; the knowledge extractor then has

$$P_2 = e(P_1,\ mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j} \cdot R, g) \tag{16}$$

Assume that the adversary generates a proof $P' = \{P_1, P_2', M', R\}$, which is different from the correct one. If the forged proof can pass the verification, the knowledge extractor has

$$P_2 = e(P_1, mpk) \cdot e(\prod_{i=1}^{c} H_2(Fid||\beta_i||\alpha)^{\alpha_i} \cdot \prod_{j=1}^{s} u_j^{M_j'} \cdot R, g) \tag{17}$$

Based on *Game* 1, we know that $P_2' = P_2$. Thus, we can conclude that

$$\prod_{j=1}^{s} u_j^{M_j} = \prod_{j=1}^{s} u_j^{M_j'} \tag{18}$$

and therefore

$$\begin{aligned} 1 &= \prod_{j=1}^{s} u_j^{\triangle M_j} \\ &= (g^a \cdot h^b)^{\sum_{j=1}^{s} \triangle M_j} \end{aligned} \tag{19}$$

Therefore, the knowledge extractor has found the solution to the DL problem $h = g^{-a/b}$ unless $b = 0 \mod p$. Therefore, the probability for this equation to fail is $1/p$, which is negligible. However, this contradicts with the DL assumption. Therefore, the theorem is proved.

4.3. Privacy Preserving Proof

Theorem 2 (Privacy Preservation)**.** *If the probability of the DV obtaining any information about the DO's identity and raw data during the audit process is negligible, then our scheme satisfies privacy preservation under Definition 2.*

Proof. The DV periodically makes data integrity challenges to the CSP in order to check whether the data stored in the cloud have their integrity. In the audit system, CSP generates a corresponding proof P for each challenge and sends it to the DV to prove that the data is well preserved. In order to prevent the DV from obtaining any information about the data and the ID of the DO, the proof generated by the CSP in our scheme is $P = \{P_1, P_2, M_1, M_2, \ldots, M_s, R\}$. The DV struggles to calculate k from $u_1, u_2, \ldots, u_s$ and $R = \prod_{j=1}^{s} u_j^{-k}$ based on the DL problem. This means that the DV cannot obtain any information about the data by challenging the same data block multiple times. In addition, the ID is hidden in $P_1 = r \cdot \prod_{i=1}^{c} H_1(ID_O)^{\alpha_i}$ by the random value r. Obviously, the DV cannot obtain the relationship between DO and data. Therefore, our scheme not only achieves the anonymity of the DO but also protects the privacy of the data in the process of data auditing. □

4.4. Detectability Proof

Theorem 3 (Detectability)**.** *If the data block stored at the CSP is damaged, the scheme proposed in this paper can effectively check it out.*

Proof. Suppose a file F consists of n blocks, where k blocks are corrupted. At the same time, assume that the TPA selects c data blocks to challenge. Let P_c denote the probability of detecting the data corruption. Then, we have

$$P_c = P\{k \geqslant 1\} = 1 - P\{k = 0\} = 1 - \prod_{i=0}^{k-1}(n-k-i)/(n-i)$$

Thus, we have

$$1 - (1 - k/n)^c < P_c < 1 - (1 - k/(n-c+1))^c$$

From the above equation, we can find that the greater the number of challenged blocks, the greater the probability of detecting corrupt blocks. For example, if 1000 of 10,000 blocks are corrupted, and 100 blocks are challenged in each data auditing, then the probability of detecting this error state is at least 99.9%. Therefore, our scheme has a very high probability of checking the data integrity. □

5. Performance Analysis

In this section, we prove the feasibility of our scheme through a theoretical analysis and experiment. In order to display the advantages of our scheme more intuitively, we compared our scheme with [21,38]. It should be noted that [38] combines the challenge algorithm with the proof generation, and we consider them separately in our analysis.

5.1. Theoretical Analysis

We compared the three schemes on the aspects of computational cost, communication cost and storage cost. Considering the running times of each algorithm, we only focus on four algorithms *TagGen*, *Challenge*, *ProofGen* and *ProofVerify* in our analysis and comparison. The remaining algorithms are executed only once in the entire scheme, so they are not analyzed. It should be noted that although the algorithm *TagGen* is only run once, considering that the function of the algorithm is to generate the tag of the data block, we choose to analyze this algorithm in the theoretical analysis. Due to the schemes of [21] being group users, we set the number of users for this schemes to 1. In the comparison of these three aspects, we ignore the parameters related to file names for a more reasonable comparison.

5.1.1. Computational Cost

Let T_p denote the computational cost of a one-time pairing operation on groups G_1 and G_2. T_{mul} and T_{exp} denote the computational cost of a one-time multiplication operation

and one time exponentiation operation, respectively. We ignore the computational cost of addition and hashing. Suppose that in our scheme, the file F is divided into n blocks, with each block containing s small data blocks. The number of data blocks in each challenge is c.

In our scheme, because Equation (4) needs to generate tags for n data blocks, algorithm *TagGen* has to run n times. The computational cost caused by one tag generation is $(s+2)T_{mul}+(s+1)T_{exp}$. Thus, the computational cost of generating tags for all data blocks in our scheme is $n((s+2)T_{mul}+(s+1)T_{exp})$. Therefore, the total computational cost of algorithm *TagGen* is $n(s+2)T_{mul}+n(s+1)T_{exp}$. Considering that we use PRP and PRF to generate challenge-related parameters, the computational cost of algorithm *Challenge* is negligible. In algorithm $ProofGen$, the proof sent by CSP to the DV is $P=\{P_1,P_2,M_1,M_2,\ldots,M_s,R\}$. The computational costs of P_1, P_2, M_j and R are $T_{mul}+T_{exp}$, $2T_p+(c+1)T_{mul}+cT_{exp}$, csT_{mul} and $sT_{mul}+sT_{exp}$, respectively. Therefore, the total computational cost of algorithm $ProofGen$ is $(c+cs+s+2)T_{mul}+(c+s+1)T_{exp}+2T_p$. In algorithm $ProofVerify$, the total computational cost is $2T_p+(3+s+c)T_{mul}+(s+c)T_{exp}$. Table 2 shows the computational cost for our scheme and the other two schemes.

As can be seen from Table 2, the computational cost of the three algorithms in our scheme is slightly higher than that of the other two schemes in the case of the same number of tag blocks. However, since our scheme collects s data blocks for signature, the number of tag generation in our scheme is far less than that of [21,38] when the number of data blocks is the same. Secondly, the computational cost of algorithm $ProofGen$ and $ProofVerity$ is slightly higher than that of [21,38] for the same reason.However, under the same numbers of proof generation and verification, our scheme can check the integrity of more data blocks. In summary, our scheme performs well in terms of computational cost.

Table 2. Comparison of computational cost.

	TagGen	**ProofGen**	**ProofVerify**
Our Scheme	$n((s+2)T_{mul}+(s+1)T_{exp})$	$(cs+c+s+2)T_{mul}+(c+s+1)T_{exp}+2T_p$	$(cs+c+s+2)T_{mul}+(c+s+1)T_{exp}+2T_p$
Yan et al. ([21])	$2n(T_{mul}+T_{exp})$	$(3c+2)T_{mul}+2cT_{exp}+2T_p$	$(c+2)T_{mul}+(c+1)T_{exp}+2T_p$
Yang et al. ([38])	$2n(T_{mul}+T_{exp})$	$2cT_{mul}+(c+2)T_{exp}+T_p$	$(c+2)T_{mul}+(c+2)T_{exp}+3T_p$

5.1.2. Communication Cost

In most PDP schemes, the communication cost consists mainly of the data uploaded by the DO to the CSP and the challenge or proof information transferred between the CSP and the DV. We analyze the communication cost of our scheme from three aspects: the upload phase, the challenge phase and the proof phase. In the upload stage, the DO sends n data blocks to CSP with corresponding tags. It is important to note that in our scheme, each data block is of the size $s|Z_p|$, and each tag is of the size $|G_1|$. Therefore, the total communication cost in the upload phase is $ns|Z_p|+n|G_1|$. The total cost of the challenge phase is $3|Z_p|$, and the total communication cost of the proof phase is $2|G_1|+|G_2|+s|Z_p|$. Table 3 shows the communication cost for our scheme and the other two schemes.

Table 3. Comparison of communication cost.

	Upload	**Challenge**	**Proof**
Our Scheme	$n\|G_1\|+ns\|Z_p\|$	$3\|Z_p\|$	$2\|G_1\|+\|G_2\|+s\|Z_p\|$
Yan et al.	$ns\|G_1\|+ns\|Z_p\|$	$3\|Z_p\|$	$\|G_1\|+\|G_2\|+\|Z_p\|$
Yang et al.	$ns\|G_1\|+ns\|Z_p\|$	$c\|Z_p\|$	$\|G_1\|+\|G_2\|+\|Z_p\|$

It can be seen from Table 3 that the communication cost of our scheme in the upload phase is higher than that of the other two schemes in the same number of tag blocks. The

main reason is that we aggregate and sign every s data blocks when generating tags, so that the number of final tags is less than the number of data blocks. In the challenge generation phase, our scheme and [21] use PRP and PRF to save communication costs between the verifier and the CSP. Finally, in the process of proof transmission, the communication cost of our scheme is slightly higher than [21,38]. In summary, our scheme has excellent performance in terms of communication cost.

5.1.3. Storage Cost

We mainly consider the storage cost of the DO, CSP and DV. From the DO's point of view, after uploading data to the CSP, the DO destroys all data locally except his own private key. Therefore, the storage overhead of the DO mainly consists of part of the private key generated by KGC and another part of the private key generated randomly by himself. Therefore, the storage cost of DO is $|G_1| + |Z_p|$. The CSP stores data blocks and their corresponding tags. In our signature scheme, there is one tag for every s small data block, and the CSP needs to store its own private key as the DO does. Therefore, the total storage cost of the CSP is $(ns + 1)|Z_p| + (n + 1)|G_1|$. The DV needs to store its own private key and trapdoor, so the storage cost is $2|G_1| + |Z_p|$. Table 4 shows the storage cost for our scheme and the other two schemes.

Table 4. Comparison of storage cost.

	DO	**CSP**	**Verifier**
Our Scheme	$\|G_1\| + \|Z_p\|$	$(n+1)\|G_1\| + (ns + 1)\|Z_p\|$	$2\|G_1\| + \|Z_p\|$
Yan et al.	$\|G_1\|$	$ns\|G_1\| + ns\|Z_p\|$	Negligible
Yang et al.	$\|G_1\|$	$ns\|G_1\| + ns\|Z_p\|$	Negligible

Table 4 shows that in our scheme, the storage cost of the DO is slightly higher than [21,38]. As with the communication cost in the upload stage, the storage cost of the CSP in our scheme is much higher than the other two schemes with the same number of tags. Due to the aggregation signature of multiple data blocks, the storage cost of the CSP in our scheme is better than [21,38] in the case of the same block. In addition, our scheme involves the authentication of the verifier's identity; the verifier must hold his own key pair, resulting in a higher storage cost than [21,38]. In summary, the storage cost of our solution performs well.

From the above three theoretical analysis and comparison experiments, it can be seen that our scheme has slight advantages compared with the other two schemes in the uploading stage regarding communication cost and the storage burden of the CSP in the storage cost. This is because our scheme signs multiple data blocks together in the tag generation algorithm instead of adopting the strategy of one data block corresponding to one tag, as in the other two schemes. As a result, the number of tags generated by our scheme is less than [21,38]. This also provides our scheme with certain advantages in these two places. In addition, in the comparison of computational cost, our scheme occupies slightly more computing resources on algorithm *TagGen* than other schemes. This is due to the fact that our scheme performs more multiplication and power operations when signing the aggregated data block. Although the computational efficiency of a single algorithm is slightly lower, the number of executions of our *TagGen* algorithm is less than other schemes when generating tags for all data blocks. Therefore, our scheme has excellent performance in theoretical analysis.

5.2. Experimental Results

To better evaluate our scheme, we implemented it using the 512-bit elliptic curve from the Pair-Based Cryptography (PBC) library [39]. The experiments were executed in ubuntu-20.04 with a Parallels Desktop 17. The configuration of the Parallels Desktop included 8 G Ram, 2 CPU and a 64 G disk. We used a MacBook Pro Laptop as the host

computer, with the settings of the macOS Big Sur 11.6 operation system, a Core Apple M1, and 16 G Ram.

We divided a 1.8GB file into 1000 data blocks according to the fixed bit length such that each data block contained 10 small data blocks. That is, $n = 1000, withs = 10$ corresponding to the theoretical analysis part. In order to better demonstrate the performance of the scheme, we used [21,38] and our own scheme to conduct the following experiments.

First, we compared the performance of the *TagGen* of the three schemes, and the results are shown in Figure 3. The horizontal axis of Figure 3 shows the number of blocks of data used to generate the tag, and the vertical axis shows the time taken. It can be seen from Figure 3 that the speed of tag generation between [21,38] is almost the same, and the speed of tag generation in our scheme is much faster than the other two schemes. Our scheme takes about 8 s to generate tags for all 10,000 data blocks, while both [21,38] need more than 25 s. In our scheme, every ten data blocks jointly generate one tag, so the total number of tags generated in our scheme is much smaller than [21,38]. In summary, the *TagGen* of our scheme is efficient.

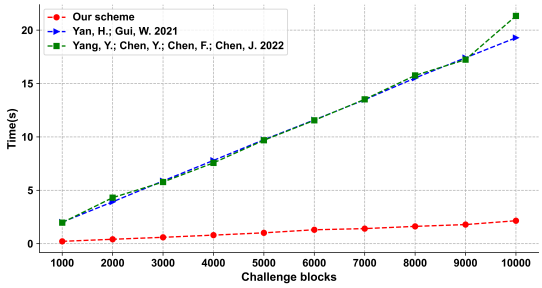

Figure 3. Computation cost of *TagGen* [21,38].

Then, we compared the execution time of the *Challenge* of the three schemes, and the results are shown in Figure 4. It can be found that as the number of *Challenge* data blocks increases, the time consumed by the *Challenge* of scheme [38] increases, and the difference in the time consumed by the challenge algorithm proposed in this paper and in [21] is negligible. Therefore, our scheme is also efficient in the challenge generation phase.

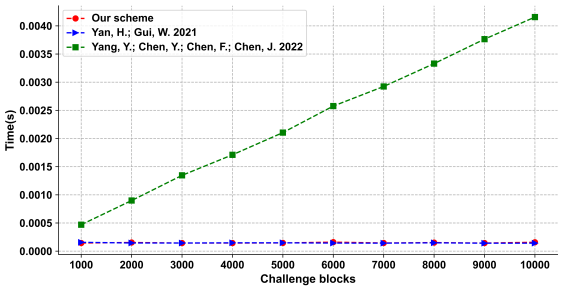

Figure 4. Computation cost of *Challenge* [21,38].

Our third experiment aimed to compare the efficiency of the *ProofGen* of the three schemes. It can be seen from Figure 5 that the efficiency of our scheme is much better than the other two schemes. With the increase in the number of data blocks, the time required to generate a correlation proof in scheme [38] increases the most. In our scheme, the time for *ProofGen* to generate a proof for all data blocks takes less than a second, while [21,38] need more than 5 s. Therefore, our scheme has a huge advantage in terms of proof generation.

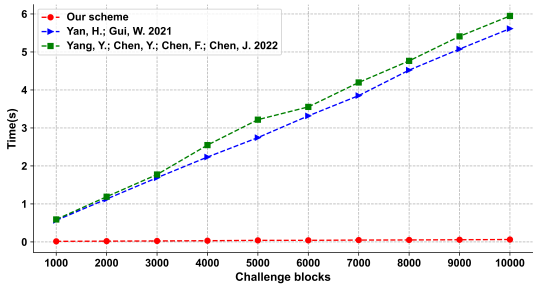

Figure 5. Computation cost of *ProofGen* [21,38].

In the last experiment, we compared the efficiency of *ProofVerify* of the three schemes, and the results are shown in Figure 6. It can be seen from the figure that the efficiency of [21] and our scheme is basically the same in the verification stage, while [38] needs more time. When auditors check the integrity of all data blocks, the verification algorithm of our scheme utilizes less than 15 s. Obviously, the *ProofVerify* of our scheme has a good performance.

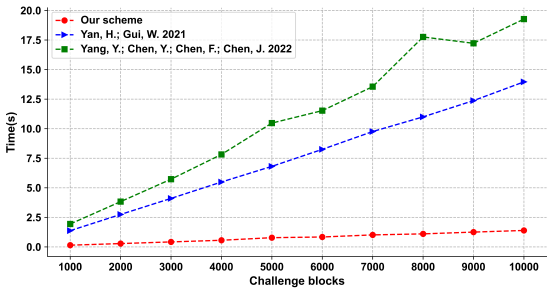

Figure 6. Computation cost of *ProofVerify* [21,38].

6. Conclusions

In this paper, we propose an effective scheme with privacy protection and a designated verifier by using certificateless signature technology. In our scheme, the data owner empowers the tag with a special trapdoor when it is generated, so that only the designated verifier can audit the integrity of the file data stored in the cloud. In addition, our scheme blinds the identity information and data blocks of the data owner during the generation phase to achieve privacy. Security analysis and experimental results show that our scheme is secure and efficient. In the future, we intend to improve the proposed scheme as much as possible on the basis of this paper. The main goal is to reduce the computational cost of the design scheme and try to integrate the dynamic update function of the data. In addition, we also intend to extend the protocol to multi-cloud and multi-copy methods and finally propose a more efficient and secure auditing scheme.

Author Contributions: Investigation, X.G.; Methodology, G.B., X.G., R.L. and W.Q.; Writing—original draft, G.B., X.G., R.L., W.Q. and Y.Z.; Writing—review & editing, G.B. and Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: This study was supported by the National Natural Science Foundation of China (No.61872284) and the Yulin Science and Technology Planning Project (CXY-2020-063).

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

References

1. Joshi, B.; Joshi, B.; Mishra, A.; Arya, V.; Gupta, A.K.; Peraković, D. A Comparative Study of Privacy-Preserving Homomorphic encryption Techniques in Cloud Computing. *Int. J. Cloud Appl. Comput. (IJCAC)* **2022**, *12*, 1–11. [CrossRef]
2. Hu, P.; Wang, Y.; Gong, B.; Wang, Y.; Li, Y.; Zhao, R.; Li, H.; Li, B. A secure and lightweight privacy-preserving data aggregation scheme for internet of vehicles. *Peer-to-Peer Netw. Appl.* **2020**, *13*, 1002–1013. [CrossRef]
3. Tewari, A.; Gupta, B.B. Secure timestamp-based mutual authentication protocol for iot devices using rfid tags. *Int. J. Semant. Web Inf. Syst. (IJSWIS)* **2020**, *16*, 20–34. [CrossRef]
4. Gaurav, A.; Gupta, B.; Peñalvo, F.J.G.; Nedjah, N.; Psannis, K. Ddos attack detection in vehicular ad-hoc network (vanet) for 5G networks. In *Security and Privacy Preserving for IoT and 5G Networks*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 263–278.
5. Fan, Y.; Lin, X.; Tan, G.; Zhang, Y.; Dong, W.; Lei, J. One secure data integrity verification scheme for cloud storage. *Future Gener. Comput. Syst.* **2019**, *96*, 376–385. [CrossRef]
6. Wu, T.; Yang, G.; Mu, Y.; Chen, R.; Xu, S. Privacy-enhanced remote data integrity checking with updatable timestamp. *Inf. Sci.* **2020**, *527*, 210–226. [CrossRef]
7. Wang, L.; Li, Y.; Yu, Q.; Yu, Y. Outsourced Data Integrity Checking with Practical Key Update in Edge-Cloud Resilient Networks. *IEEE Wirel. Commun.* **2022**, *29*, 56–62. [CrossRef]
8. Ding, Y.; Li, Y.; Yang, W.; Zhang, K. Edge data integrity verification scheme supporting data dynamics and batch auditing. *J. Syst. Archit.* **2022**, *128*, 102560. [CrossRef]
9. Tian, H.; Nan, F.; Jiang, H.; Chang, C.C.; Ning, J.; Huang, Y. Public auditing for shared cloud data with efficient and secure group management. *Inf. Sci.* **2019**, *472*, 107–125. [CrossRef]
10. Huang, P.; Fan, K.; Yang, H.; Zhang, K.; Li, H.; Yang, Y. A Collaborative Auditing Blockchain for Trustworthy Data Integrity in Cloud Storage System. *IEEE Access* **2020**, *8*, 94780–94794. [CrossRef]
11. Yoosuf, M.S.; Anitha, R. LDuAP: Lightweight dual auditing protocol to verify data integrity in cloud storage servers. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 3787–3805. [CrossRef]
12. 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]
13. 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]
14. Wang, Q.; Wang, C.; Ren, K.; Lou, W.; Li, J. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.* **2011**, *22*, 847–859. [CrossRef]
15. Garg, N.; Bawa, S.; Kumar, N. An efficient data integrity auditing protocol for cloud computing. *Future Gener. Comput. Syst.* **2020**, *109*, 306–316. [CrossRef]
16. 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.* **2021**, *9*, 1408–1421. [CrossRef]
17. Peng, S.; Zhou, F.; Li, J.; Wang, Q.; Xu, Z. Efficient, dynamic and identity-based Remote Data Integrity Checking for multiple replicas. *J. Netw. Comput. Appl.* **2019**, *134*, 72–88. [CrossRef]
18. Li, J.; Yan, H.; Zhang, Y. Identity-Based Privacy Preserving Remote Data Integrity Checking for Cloud Storage. *IEEE Syst. J.* **2021**, *15*, 577–585. [CrossRef]
19. Yang, Y.; Chen, Y.; Chen, F. A Compressive Integrity Auditing Protocol for Secure Cloud Storage. *IEEE/ACM Trans. Netw.* **2021**, *29*, 1197–1209. [CrossRef]
20. Tian, J.; Wang, H.; Wang, M. Data integrity auditing for secure cloud storage using user behavior prediction. *Comput. Secur.* **2021**, *105*, 102245. [CrossRef]
21. 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]
22. Yan, H.; Li, J.; Zhang, Y. Remote Data Checking with a Designated Verifier in Cloud Storage. *IEEE Syst. J.* **2020**, *14*, 1788–1797. [CrossRef]
23. Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Kissner, L.; Peterson, Z.; Song, D. *Provable Data Possession at Untrusted Stores*; ACM Press: New York, NY, USA, 2007; p. 598. [CrossRef]
24. Juels, A.; Kaliski, B.S. *Pors*; ACM Press: New York, NY, USA, 2007; p. 584. [CrossRef]
25. Yan, H.; Li, J.; Han, J.; Zhang, Y. A Novel Efficient Remote Data Possession Checking Protocol in Cloud Storage. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 78–88. [CrossRef]
26. Sun, Y.; Liu, Q.; Chen, X.; Du, X. An Adaptive Authenticated Data Structure with Privacy-Preserving for Big Data Stream in Cloud. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3295–3310. [CrossRef]
27. Zhu, H.; Yuan, Y.; Chen, Y.; Zha, Y.; Xi, W.; Jia, B.; Xin, Y. A Secure and Efficient Data Integrity Verification Scheme for Cloud-IoT Based on Short Signature. *IEEE Access* **2019**, *7*, 90036–90044. [CrossRef]
28. 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]
29. Shang, T.; Zhang, F.; Chen, X.; Liu, J.; Lu, X. Identity-Based Dynamic Data Auditing for Big Data Storage. *IEEE Trans. Big Data* **2021**, *7*, 913–921. [CrossRef]

30. Zhao, C.; Xu, L.; Li, J.; Wang, F.; Fang, H. Fuzzy Identity-Based Dynamic Auditing of Big Data on Cloud Storage. *IEEE Access* **2019**, *7*, 160459–160471. [CrossRef]
31. 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.* **2019**, *14*, 331–346. [CrossRef]
32. Li, Y.; Zhang, F. An efficient certificate-based data integrity auditing protocol for cloud-assisted WBANs. *IEEE Internet Things J.* **2021**. [CrossRef]
33. Wang, F.; Xu, L.; Choo, K.K.R.; Zhang, Y.; Wang, H.; Li, J. Lightweight Certificate-Based Public/Private Auditing Scheme Based on Bilinear Pairing for Cloud Storage. *IEEE Access* **2020**, *8*, 2258–2271. [CrossRef]
34. Du, H.; Wen, Q.; Zhang, S.; Gao, M. A new provably secure certificateless signature scheme for Internet of Things. *Ad Hoc Netw.* **2020**, *100*, 102074. [CrossRef]
35. Li, J.; Yan, H.; Zhang, Y. Certificateless public integrity checking of group shared data on cloud storage. *IEEE Trans. Serv. Comput.* **2018**, *14*, 71–81. [CrossRef]
36. Zhou, L.; Fu, A.; Yang, G.; Wang, H.; Zhang, Y. Efficient Certificateless Multi-Copy Integrity Auditing Scheme Supporting Data Dynamics. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 1118–1132. [CrossRef]
37. 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]
38. Yang, Y.; Chen, Y.; Chen, F.; Chen, J. Identity-Based Cloud Storage Auditing for Data Sharing with Access Control of Sensitive Information. *IEEE Internet Things J.* **2022**, *9*, 10434–10445. [CrossRef]
39. Lynn, B.; Shacham, H.; Steiner, M.; Cooley, J.; Figueiredo, R.; Khazan, R.; Kosolapov, D.; Bethencourt, J.; Miller, P.; Cheng, M.; et al. PBC Library. Available online: http://crypto.stanford.edu/pbc (accessed on 20 May 2020).