IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 9, 2020, accepted November 19, 2020, date of publication November 23, 2020,
date of current version December 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3039981

Stateless Cloud Auditing Scheme for Non-Manager Dynamic Group Data With Privacy Preservation

XIAODONG YANG[1], (Member, IEEE), MEIDING WANG[1], XIUXIU WANG[1], GUILAN CHEN[1], AND CAIFEN WANG[1,2]

[1]College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China
[2]College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China

Corresponding author: Xiaodong Yang (y200888@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61662069 and Grant 61562077, in part by the China Postdoctoral Science Foundation under Grant 2017M610817, in part by the Science and Technology Project of Lanzhou City under Grant 2013-4-22, and in part by the Foundation of Northwest Normal University under Grant NWNU-LKQN-14-7.

ABSTRACT As one of the core services of cloud computing, cloud storage could satisfy various storage and management requirements caused by the growth of data. Considering the complexity and uncontrollability of the cloud storage environment, many cloud auditing schemes were presented to assure the integrity of data in the cloud. However, most existing schemes have security risks, such as identity privacy and data privacy disclosure, authority abuse of group managers and collusion attacks during user revocation. To solve these problems, we propose a stateless cloud auditing scheme for non-manager dynamic group data with privacy preservation. The proposed scheme not only realizes user identity privacy preservation but also preserves data privacy security with the random masking technique. Unlike other solutions, our scheme allows t group users to trace the user's identity cooperatively without group managers, which eliminates authority abuse of group managers and provides non-frameability. Meanwhile, utilizing the concept of Shamir Secret Sharing, our scheme divides the re-signing process into several parts to resist collusion attacks during user revocation. By the designed binary tree, group users could trace dynamic data changes and recover the latest data when existing data are damaged. Besides, both users and the third-party auditor (TPA) are stateless in our scheme; that is, they no need to maintain data index information during cloud auditing. Our scheme also achieves mutual supervision between users and cloud service providers (CSPs), which ensures data are non-repudiation on both parties. Furthermore, we construct an efficient incentive for data visitors by using the blockchain technology and design a secure data sharing model to guarantee that data owners control their data ownership. Certificateless cryptography assures that the proposed scheme avoids certificate management and key escrow problems. Finally, security analysis and performance evaluation show that our scheme is secure and efficient.

INDEX TERMS Certificateless, cloud auditing, dynamic data, non-manager, privacy preservation, stateless.

I. INTRODUCTION

Cloud storage is a crucial part of the cloud computing platform, which makes individuals and groups enjoy virtualized infrastructure while avoiding paying huge expenses. Due to many advantages of cloud storage, more and more users chose to store their data to some well-known CSPs, such as iCloud and Google Docs [1]. Although these CSPs promise to provide a safe and reliable environment to users, byzantine failures, malicious external and internal attacks [2], [3] may still affect the integrity of data in the cloud. Since users lose direct control of data, they cannot know the status of the data in the cloud. Therefore, to assure the integrity of the data in the unstable cloud, it is crucial to develop cloud auditing.

In recent years, many cloud auditing schemes based on the traditional public key cryptography (PKC) have been proposed. In these schemes, the user's identity and public key are bound together by a digital certificate. However, the distribution and management of certificates bring a heavy computational overhead. To address this concern, many researchers

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Imran Tariq.

focused on studying identity-based cryptography (IBC) cloud auditing schemes. The public keys of users are generated with their unique identities, and users' private keys are generated by the private key generation center (PKG). Unfortunately, once PKG is compromised, it can easily impersonate any user to forge tags without being discovered. In contrast, certificateless cryptography is a good choice, where the user's private key consists of a secret value and a partial key. The secret value is chosen by the user, and the partial key is generated by the key generation center (KGC), which eliminates the key escrow problem. Besides, the user's public key is generated by himself/herself, so certificateless cryptography also avoids the certificate management problem.

In certificateless settings, most cloud auditing schemes only focus on personal data [4]–[6]. When users want to share data with others in the group, some new issues appear. For instance, the TPA could find which user is more important in the group, and which data are more critical after several audits. Meanwhile, frequent data auditing may bring more attacks, such as address tracking. Therefore, it is vital to achieve identity anonymous of group users during data auditing. However, anonymity protects the identity privacy of group users while also leading to other risks. For example, when malicious group users upload illegal data or modify shared data for their interests, the property of anonymity offers a protective umbrella for them and makes them escape punishment. Aim at this problem, researchers proposed a series of cloud auditing schemes with traceability. Unfortunately, some of these schemes rely on one group manager with hugely high permissions. If this group manager discloses private information, it will bring severe threats to the identity privacy security of group users. There are also some schemes that require multiple group managers to track misbehaved users' identities cooperatively. Although they provide non-frameability, a lot of extra overheads are also generated because of the joining of group managers. Therefore, it is still an open challenge to design a cloud auditing scheme for identity privacy preservation and identity traceability.

Notably, in a secure cloud auditing scheme, the issue of data content disclosure in the process of auditing should be avoided. As the TPA is not full-trusted [7], he/she could collect a suffcient number of linear combinations from auditing information to obtain the sampled shared data content by solving linear equations [8]. To protect the data privacy, some shared cloud data auditing schemes use the random masking technology [9] or zero-knowledge privacy technology [10] to prevent the TPA from getting any information on the shared data. Unfortunately, in these schemes, the generation of tags, the update of data, and the verification of auditing proofs involve the data index information. The group users and the TPA need to maintain a large data index table or index-hash table, which significantly increases the computation costs. Therefore, ensuring data privacy security, and achieving group users and the TPA stateless are both essential to cloud auditing schemes.

Meanwhile, considering the group is dynamic, users should be able to revoke from the group at any time. Since the tags of shared data are generated with users' private keys, all tags of the revoked user need to transform into the tags of an existing group user. In traditional cloud auditing schemes, an existing group user is required to download all data of the revoked user, re-sign these data, and send new tags to the CSP. These operations produce many computation overheads. To overcome these shortcomings, some new cloud auditing schemes [11]–[14] that support user revocation are proposed. In these schemes, the CSP communicates with the revoked user to transform his/her all tags into the tags of one existing group user by using the re-signing key. However, the revoked user and the CSP could obtain private keys of existing group users by colluding. Therefore, how to ensure efficient and safe revocation of group users becomes an urgent issue to be solved in the cloud auditing schemes for shared data.

Furthermore, cloud storage is not only a data warehouse but also should ensure that group users update data dynamically according to different application purposes. For the sake of dynamic data integrity checking, researchers proposed data structures based on the index hash table (IHT) [12]–[15] and Merkle Hash Tree (MHT) [16]–[18] to support dynamic data operations. Unfortunately, these data structures can only record the latest data and the corresponding tags, making it impossible for group users to track the changes of data. If the existing data are lost or corrupted, users cannot recover them from the records. Thus, the issues of data traceability and data recoverability also should be taken into consideration.

At present, there is usually a lack of efficient mutual supervision between the CSP and users. On the one hand, to get compensation from the CSP, group users may falsely claim that their data in the cloud are lost. On the other hand, the CSP may only save old data for users and refuse to keep the updated data. In this case, it is difficult to determine which party tells the truth. Therefore, unsupervised data uploading is another important problem that should be solved.

So far, few schemes consider that the data owners may lose control of their data once the data are copied by other users in the process of data sharing. If other group users trade the copied shared data for profit, the rights of the data owners would be seriously violated. Meanwhile, during shared data accessing, data visitors should be rewarded according to the number of times that they access shared data. Therefore, security and practicality also cannot be ignored during data sharing and data accessing.

A. OUR CONTRIBUTIONS

In this paper, a stateless cloud auditing scheme for non-manager dynamic group data with privacy preservation is proposed. We summarize significant contributions as follows.

(1) We present an efficient and secure certificateless cloud auditing scheme for shared data, which not only avoids the certificate management of PKC but also eliminates key escrow problem of IBC.

(2) Our scheme could satisfy multi-levels of privacy preservation, include user identity privacy preservation and data content privacy preservation. Specifically, the TPA cannot obtain group users' identities information and shared data content from tags of shared data and auditing proof.

(3) Our scheme could realize identity traceability of group users without any group manager while assuring identity anonymous of users. Each group user in the proposed scheme is assigned the equal power to manage the group. Leveraging the Lagrange interpolation, at least t valid group users can trace the identity of the misbehaved user from the tags, which guarantees non-frameability of the scheme and avoids authority abuse of group managers.

(4) Our scheme achieves group users and the TPA stateless by introducing a new entity called cloud partner (CP), where the CP only stores a small amount of metadata. This method makes group users and the TPA have no more need to maintain complex data index tables and reduces overheads of data auditing and dynamic data updates.

(5) Our scheme realizes efficient and secure user revocation. Unlike the traditional method, the existing user does not need to perform tag processing operations. Based on the concept of Shamir secret sharing, the re-signing process of our scheme is divided into several parts, and are deployed to the CP. The method prevents the CSP from transforming tags between any two group users and makes the collusion attacks between the revoked user and the CSP impossible.

(6) We design a data structure based on the binary tree, which not only could support the dynamic data auditing but also achieves traceability and recoverability of data. The group users could easily find the previously stored data and recover these data content by the designed binary tree when the existing data are damaged.

(7) Our scheme achieves mutual supervision between group users and the CSP. In the process of interaction, the user and the CSP collaborate to generate a receipt of the upload data, which ensures the data are non-repudiation to both parties.

(8) We design a secure shared data accessing model, which provides data visitors with an access interface instead of directly sharing data to them. This method offers data owners with protection of the ownership and the control of data. Meanwhile, we use the blockchain technology to record the times that data visitors access the shared data, and achieves effective incentive for data visitors according to these recordings.

(9) The proposed scheme is proved to have strong security in the random model, which could resist two types of attacks of the certificateless environment and satisfy many security requirements. The performance analysis shows that our scheme is more efficient than other related schemes in communication and computation costs.

B. RELATED WORKS

To provide better cloud storage services to users, Ateniese *et al.* [19] presented the first provable data possession (PDP) model, which allows users to check the integrity of data without retrieving all files in the cloud. Then Juels and Kaliski [20] presented the proofs of retrievability (POR) model, which could generate the proof that the verifier retrieves data. Based on PDP and POR models, more cloud auditing schemes were proposed to satisfy different application requirements.

We remark that most cloud auditing schemes are based on PKC [14], [21]. In these schemes, the user needs to check the validity of the certificate every time before using his/her public key, which produces much computational overhead. For the case, IBC is adopted by many cloud auditing schemes [22], [23]. For instance, Yu *et al.* [24] presented an IBC cloud auditing scheme by using RSA signature technology, which realizes variable-size file blocks and cloud data public auditing. Wang *et al.* [25] presented a proxy-oriented IBC cloud auditing scheme, which could upload and audit the cloud data for managers. Although IBC avoids certificate management, it is not the best choice for auditing data integrity due to the inherent key escrow drawback. Wang *et al.* [26] first presented the certificateless cloud auditing scheme, which eliminates the key escrow problem. Li *et al.* [27] presented a certificateless shared cloud data auditing scheme, but it cannot protect the privacy security of cloud data and user identity privacy security against the TPA.

Besides safe key management, the property of identity anonymity is also important for group users. Wang *et al.* [15] proposed the first user identity protection mechanism "Oruta" supporting cloud data auditing and used the ring signature technology to construct a cloud auditing scheme with user identity privacy preservation. He *et al.* proposed a privacy preservation cloud auditing scheme for group users [28]. In this scheme, the tags of every user are transformed into the tags of the TPA, which assures the identity security of group users against the TPA. Later, Wu *et al.* [29] proposed a certifcateless cloud auditing scheme with privacy preservation for group users. However, none of the above three schemes supports the group user identity tracking. To solve the problem, Wang *et al.* [12] proposed a cloud auditing scheme "Knox" by using group signature technology, which realizes the identity traceability of misbehaved users by the group manager. Nevertheless, this method may cause that the innocent group user is framed and the malicious user is harbored because of the high authority of the group manager. Fu *et al.* [30] presented a cloud auditing scheme with traceability, which requires multiple group managers to work together to disclose malicious user identity. Despite scheme [30] overcomes the problem of frameability, such method of centralized control is still undesirable in some applications, such as a group is managed jointly by multiple users.

When it comes to privacy preservation, data privacy protection is also a vital property to shared cloud data auditing. Early on, most cloud auditing schemes had the data privacy problem, because the challenged data were aggregated into the linear combination to be one part of the auditing proof and the TPA easily get the data content from the auditing

proof. For example, Yang *et al.* [31] indicated the fact that the scheme "Panda" [14] cannot resist the auditing proof forgery attack and may cause the data content leakage. Yu *et al.* [32] proposed a cloud auditing scheme with the perfect data privacy preservation relying on IBC. Zhu *et al.* [33] also proposed a cloud auditing scheme with data privacy preservation that achieves data dynamics utilizing an index hash table. Nevertheless, in the above schemes, group users and the TPA are required to maintain an index related table, which involves heavy computational and storage burden. Zhao *et al.* [34] proposed a group users stateless cloud auditing scheme with data privacy preservation, which only considers the stateless of users, and ignores the stateless of the TPA. What's more, the scheme also cannot support the user revocation.

To realize that users could revoke from the group flexibly and efficiently, many schemes [13], [14] are proposed. In these schemes, the CSP transforms the tags of the revoked user into the tags of an existing group user. Although the method does not affect existing group users, it is vulnerable to collusion attacks. Wang *et al.* [14] proposed a cloud auditing scheme that supports efficient user revocation. However, the CSP must know the re-signing keys of both users in advance, which bring some security flaws. For instance, a malicious CSP could arbitrarily specify a user in the group to receive the revoked user's tags. The CSP and the revoked user could also launch collusion attacks to get private keys of existing users.

Since data stored in the cloud may be updated frequently for various application requirements, cloud auditing schemes also need to consider the dynamic nature of cloud data. However, the previous cloud auditing schemes could only verify the integrity of static data in the cloud. To address this issue, Zhang *et al.* [18] presented an MHT-based cloud auditing scheme to realize the auditing for dynamic data in the cloud. Later, a modified MHT cloud auditing scheme [35] with each node containing two values were proposed, which reduces the computational complexity of finding leaf nodes. However, the MHT-based cloud auditing schemes still have some severe overhead problems. Zhu *et al.* [33] proposed another data structure IHT that supports data dynamics, but it cannot achieve traceability and recoverability of data. In 2014, Mo *et al.* [36] presented an MHT-based data possession verification scheme with non-repudiation. However, the scheme also does not support privacy preservation and TPA auditing.

In addition, to prevent data owners from losing control of their data, Huang *et al.* [37] proposed a privacy-preserving cloud auditing scheme with secure data sharing, but the scheme does not realize that the data are non-repudiation to group users and the CSP. Meanwhile, the incentive of the scheme is designed for data signers, which ignores data visitors.

II. PRELIMINARIES

In this section, we introduce some preliminaries that used in this paper, including bilinear pairing, hardness assumptions, threshold secret sharing and blockchain technology.

A. BILINEAR PAIRING

Let G_1 and G_2 be cyclic groups with the same prime order p. g is the generator of G_1. The bilinear map $e : G_1 \times G_1 \rightarrow G_2$ satisfies the following conditions.

(1) Bilinearity: For any $g_2, g_3 \in G_1$, $r, s \in Z_q^*$, there is $e\left(g_2^r, g_3^s\right) = e\left(g_2, g_3\right)^{rs}$.

(2) Non-degeneracy: For any $g_2, g_3 \in G_1$, there is $e\left(g_2, g_3\right) \neq 1$.

(3) Computability: For any $g_2, g_3 \in G_1$, $e\left(g_2, g_3\right)$ can be calculated.

B. HARDNESS ASSUMPTIONS

Definition 1 (CDH Problem): Suppose G_1 is a cyclic group, g is the generator of G_1. Given a tuple $\left(g, g^a, g^b\right)$ containing unknown elements $a, b \in Z_q^*$. It is difficult to calculate $g^{ab} \in G_1$.

Definition 2 (CDH Hypothesis): For any probabilistic polynomial time algorithm $\mathcal{C}$, the probability of solving the CDH problem in G_1 is defined as:

$$Adv_{CDH}\left(C\right) = \Pr\left[\left\{g^{ab}\right\} \longleftarrow C\left(g, g^a, g^b\right)\right].$$

If $Adv_{CDH}\left(C\right)$ is negligible, it is difficult to solve the CDH problem.

C. SHAMIR THRESHOLD SCHEME

In 1979, Shamir first proposed a (t, n) threshold scheme based on polynomial Lagrange interpolation formula [38]. The scheme can distribute a secret U among n users of a group, and each group user is assigned a share of U. The secret U reconstruction requires at least t group users. The steps of the (t, n) threshold scheme are as follows.

(1) Secret division: First, the secret distributor D selects $t-1$ elements $a_i \in Z_q^* \left(i = 1, \ldots t-1\right)$ randomly to construct a polynomial $L\left(x\right) = U + a_1 x + \cdots + a_{t-1} x^{t-1}$, where $L\left(0\right) = U$. Second, D selects random values $x_k \in Z_q^*, k = 1, \ldots, n$, and computes $y_k = L\left(x_k\right)$. Finally, D sends $\left(x_k, y_k\right)$ to the group user u_k in secret. The polynomial is confidential and should be destroyed.

(2) Secret recovery: Suppose any t group users restore the secret U together. First, t group users offer their shares $\left(x_1, y_1\right), \left(x_2, y_2\right), \ldots, \left(x_{t-1}, y_{t-1}\right)$. Second, t group users compute $f_l\left(x\right) = \prod_{\substack{j=1 \\ j \neq l}}^{t} \frac{x - x_j}{x_l - x_j}$, and get the polynomial $L\left(x\right) = f_l\left(x\right) \cdot \sum_{l=1}^{t} y_l$. Finally, the constant term $L\left(0\right)$ of the polynomial $L\left(x\right)$ is the secret U to be recovered.

D. BLOCKCHAIN TECHNOLOGY

Blockchain is a list of orderly records linked together by blocks, which is essentially a decentralized database.

According to the degree of network centralization, blockchain can be divided into three modes: public blockchain, consortium blockchain and private blockchain. The public blockchain is completely decentralized and

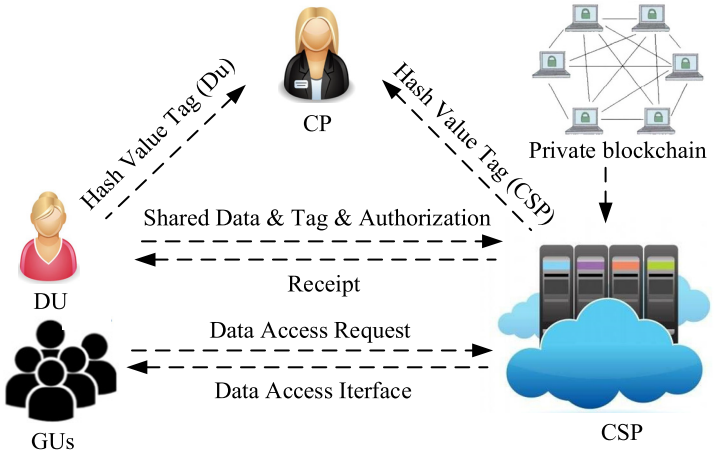

FIGURE 1. Data accessing model.

permissionless. Users could visit any node in the public blockchain. The consortium blockchain is a partially decentralized blockchain, which is usually jointly managed by multiple organizations. The users who authorized by the organization could access the consortium blockchain. The private blockchain is a fully centralized blockchain with tamper resistance. Meanwhile, the access rights are controlled by a central authority.

In our scheme, we use a private blockchain to record the shared data accessing information of group users, which is only open to group users and the CSP.

III. SYSTEM MODEL AND SECURITY MODEL

In this section, we introduce system model, the definition of our scheme and security model.

A. SYSTEM MODEL

The system model in this paper mainly consists of two parts: data accessing and data auditing. The former is the model that group users access shared data, and the latter is the model that the TPA checks the integrity of shared data in the cloud. As Figure 1 shows, the data accessing model mainly includes four entities: the data uploader (DU), the group users (GUs), CP and CSP. The specific interaction process is as follows: Firstly, the DU generates the hash value, the tags of hash value and the data, and sends the hash value tag to the CP, the shared data with the tag and the authorization to the CSP. Secondly, after verifying the identity of the DU, the CSP generates and sends the tag of the hash value to the CP, the receipt to the DU. Thirdly, GUs send requests to the CSP to access the shared data. The requests contain the identity information of the visitors and the accessed data identity. Since the blockchain technology has the property of the tamper resistance, it could record the identity of the data visitor and the access time accurately. The DU could obtain rewards according to these accessing recordings kept in the private blockchain. Finally, after receiving requests, the CSP shares the data access interface with GUs, which assures the benefits of the DU.

As Figure 2 shows, the data auditing model mainly includes five entities: KGC, GUs, CP, CSP and TPA. The specific interaction process is as follows: Firstly, GUs generate their private keys and public keys utilizing partial keys distributed from the KGC, and compute re-signing key shares and send them to the CP. Secondly, GUs send auditing requests that are used to check the integrity of the data in the cloud to the TPA. Thirdly, the TPA generates the challenge for GUs and sends them to the CSP. After receiving the challenge, the CSP generates the proof and sends it to the TPA. Finally, the TPA gets the hash value from the CP to verify the correctness of the proof, and sends a response to GUs.

(1) **KGC:** It is a semi-trusted third-party entity who could output public parameters and system master key, and generate partial keys for every group user.

(2) **GUs:** They are users of the group who have a lot of data. In order to reduce the burden of data storage and maintenance, they store and share their data in the cloud. They could also access and modify the shared data in the cloud.

(3) **DU:** It is the data uploader and a member of GUs, whose responsibility is uploading the shared data.

(4) **CSP:** It is a third-party entity to coordinate and manage several cloud servers to provide computation resources and the shared data storage service.

(5) **TPA:** It is the third-party auditor that only serves GUs and has ability to audit the integrity of the shared data for GUs.

(6) **CP:** It is a reliable entity, whose responsibility is storing the hash value of shared data, publishing hash value to TPA for auditing, and helping GUs revoke from the group.

B. DEFINITION OF OUR SCHEME

Our scheme includes ten algorithms, namely system setup algorithm and partial key generation algorithm run by KGC,

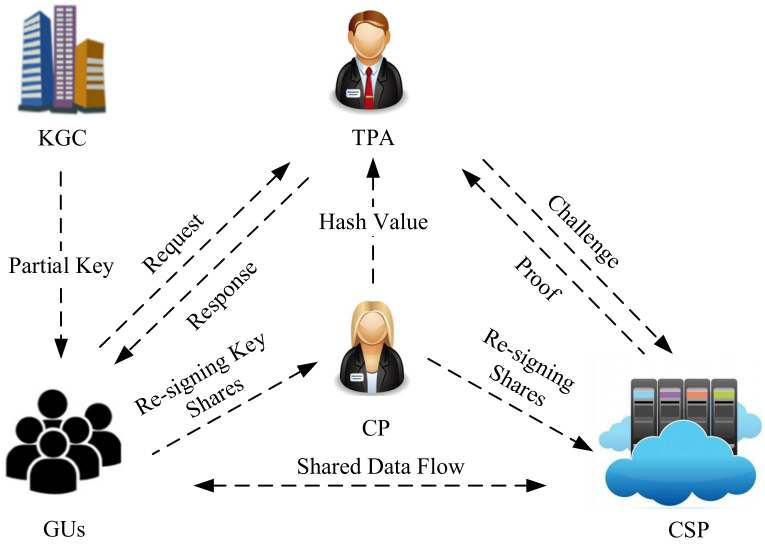

FIGURE 2. Data auditing model.

secret value generation algorithm, public key generation algorithm, share generation algorithm and tag generation algorithm run by a group user, challenge generation algorithm and proof verify algorithm run by the TPA, and tag verify algorithm, proof generation algorithm run by the CSP. They are represented by Setup, PartialKeyGen, SecretValueGen, PublicKeyGen, ShareGen, TagGen, TagVerify, ChallGen, ProofGen, and ProofVerify. These algorithms are described as follows.

Setup $(\Delta) \rightarrow \alpha, params$: Input a security parameter Δ, it returns the parameters *params*, the system master key α.

PartialKeyGen $(params, \alpha, ID_i) \rightarrow D_i$: Input *params*, α and the identity ID_i, it returns the partial key D_i.

SecretValueGen $(params, ID_i) \rightarrow \gamma_i$: Input *params* and ID_i, it returns the secret value γ_i.

PublicKeyGen $(params, \gamma_i) \rightarrow pk_i$: Input *params* and γ_i, it returns the public key pk_i.

ShareGen $(params, \gamma_i, pk_k) \rightarrow (x_k, y_k, z_k), F_i$: Input *params*, γ_i and the public keys pk_k $(k = 1, \ldots, n, k \neq i)$ of group users, it returns the sharing share $(x_k, y_k, z_k)_{k=1,\ldots,n,k\neq i}$ and F_i.

TagGen $(params, D_i, \gamma_i, pk_i, m, w, T) \rightarrow \sigma, N^*, \mathcal{T}^*$: Input *params*, D_i, γ_i, pk_i, the data m, the index w and the time stamp T, it returns the tag σ, the hash value tag N^* signed by u_i and the authorization T^*.

TagVerify $(params, \mathrm{T}^*, \sigma) \rightarrow T', R$, "1"/"0": Input *params*, T^* and σ, it returns the receipt T', the hash value tag R signed by CSP. It also returns "1" or "0". "1" means the validation succeed, and "0" means the validation failed.

ChallGen $(params, j_{\max}) \rightarrow chal$: Input *params* and the maximum index $j_{\max}$, it returns the challenge *chal*.

ProofGen $(params, chal) \rightarrow proof$: Input *params* and *chal*, it returns the proof *proof*.

ProofVerify $(params, \mathcal{A}, \mathrm{B}) \rightarrow$ "1", "0": Input *params*, the data set $\mathcal{A}$ and the tag set B, it returns "1" or "0". "1" means the validation succeed, and "0" means the validation failed.

C. SECURITY MODEL

In the certificateless environment, we introduce two adversaries to demonstrate the security of the proposed scheme. $\mathcal{A}_{\mathrm{I}}$ is a dishonest adversary who could replace any user's public key in the group with other value, although he/she cannot get ∂. $\mathcal{A}_{\mathrm{II}}$ is a curious adversary whose ability is accessing ∂ instead of replacing the public key of any group user.

In order to prove our scheme could be secure against two types of adversaries, we define some oracles at first.

Partial Key Generation Oracle O^d (ID^*): The adversary inputs the identity ID^*, it outputs D_{ID^*} as the partial key of ID^*.

Secret Value Generation Oracle O^s (ID^*): The adversary inputs the identity ID^*, it outputs γ_{ID^*} as the secret value of ID^*.

Public Key Oracle O^p (ID^*): The adversary inputs the identity ID^*, it outputs pk_{ID^*} as the public key of ID^*.

Public Key Replace Oracle O^r $\left(ID^*, pk'_{ID^*}\right)$: The adversary inputs the identity ID^* and the public key pk'_{ID^*}, it replaces the corresponding public key of ID^* with pk'_{ID^*}.

Tag Generation Oracle O^t (ID^*, w^*, m^*): The adversary inputs the identity ID^*, the index w^* and the data m^*, it outputs the tag of the data m^* on ID^* with the public key pk'_{ID^*}.

Next, we define the security model against adversary $\mathcal{A}_{\mathrm{I}}$. The specific process between the adversary $\mathcal{A}_{\mathrm{I}}$ and the challenger $\mathcal{C}$ is described as follows.

Setup: $\mathcal{C}$ performs the system setup algorithm Setup, sends public parameter *param* to $\mathcal{A}_{\mathrm{I}}$, and remains the system master key α secretly.

Queries: $\mathcal{A}_{\mathrm{I}}$ inquires the partial key generation oracle O^d, secret value generation oracle O^s, public key oracle O^p,

public key replace oracle O^r, and tag generation oracle O^t. $\mathcal{C}$ generates the responses for these queries.

Forgery: $\mathcal{A}_{\text{I}}$ returns the forged tag σ' with the public key $pk_{ID'}$ of the identity ID' on data m'.

If all the following conditions are true, $\mathcal{A}_{\text{I}}$ wins the game.

(1) $\mathcal{A}_{\text{I}}$ never asks for the partial key generation oracle O^d of the identity ID'.

(2) $\mathcal{A}_{\text{I}}$ never asks for the tag generation oracle O^t on data m' of the identity ID'.

(3) $\mathcal{A}_{\text{I}}$ generates the forged tag σ' that is valid.

Last, we define the security model against adversary $\mathcal{A}_{\text{II}}$. The specific process between the adversary $\mathcal{A}_{\text{II}}$ and the challenger $\mathcal{C}$ is described as follows.

Setup: $\mathcal{C}$ performs the system setup algorithm Setup, and sends public parameter *param* and the system master key ∂ to $\mathcal{A}_{\text{II}}$.

Queries: $\mathcal{A}_{\text{II}}$ inquires the secret value generation oracle O^s, public key oracle O^p, and tag generation oracle O^t. $\mathcal{C}$ generates the responses for these queries.

Forgery: $\mathcal{A}_{\text{II}}$ returns the forged tag σ' of the identity ID' on data m'.

If all the following conditions are true, $\mathcal{A}_{\text{II}}$ wins the game.

(1) $\mathcal{A}_{\text{II}}$ never asks for the secret value generation oracle O^s of the identity ID'.

(2) $\mathcal{A}_{\text{II}}$ never asks for the tag generation oracle O^t of data m' with the identity ID'.

(3) $\mathcal{A}_{\text{II}}$ generates the forged tag σ' that is valid.

IV. THE PROPOSED SCHEME

In this section, we introduce the construction and properties of our scheme in detail. The main notations used in the proposed scheme are listed in Table 1.

A. CONSTRUCTION OF SCHEME

To realize the non-frameability and high efficiency, we design a shared cloud data auditing scheme without group managers. Note that the group with a threshold t is pre-defined before the original user shares his/her data in the cloud, and the initial group users are decided by the original user. Later, the group is managed by all group users during data sharing.

We suppose that our scheme has n group users u_i $(1 \le i \le n)$ and Z data $m_j \in Z_q^*$ $(1 \le j \le Z)$. The specific description is as follows.

(1) **Setup:** The KGC generates public parameters and the system master key by performing the following steps.

- The KGC selects the bilinear mapping $e : G_1 \times G_1 \to G_2$, where G_1 and G_2 are two cyclic groups with prime number p, and g is the generator of G_1.
- The KGC selects two hash functions $H_1 : \{0, 1\}^* \to G_1$ and $H_2 : \{0, 1\}^* \to G_1$.
- The KGC selects the system master key $\alpha \in Z_q^*$, and computes $g_1 = g^\alpha$.
- The KGC saves α in secret, and publishes public parameters $params = \{G_1, G_2, H_1, H_2, g, g_1\}$.

TABLE 1. Notations and descriptions of scheme.

Notations	Descriptions
G_1, G_2	The cyclic groups with prime number p
g	The generator of G_1
H_1, H_2	The hash functions
α	The system master key
D_i	The partial key of u_i
γ_i	The secret value of u_i
pk_i	The public key of u_i
$L(x), P(x)$	The random polynomials
(x_k, y_k, z_k)	The shares that sends to u_k
w_j	The name of shared data m_j
N_j	The hash value of w_j
σ_j	The tag of m_j
N_j^*	The hash value tag of N_j signed by u_i
T_j^*	The authorization
(ξ, g^ξ)	The private and public key of CSP
T_j'	The receipt between u_i and CSP for m_j
R_j	The hash value tag of N_j signed by CSP
$\{j, v_j\}$	The challenge information with index of j
$\{\Gamma, \lambda, \overline{\sigma_j}, \overline{ID}, \overline{PK}\}$	The proof of the challenge

(2) **PartialKeyGen:** The user u_i sends the identity ID_i to KGC. The detailed steps of partial key generation are as follows.

- The KGC computes $D_i = H_1 (ID_i)^\alpha$ as the partial key of u_i.
- The KGC sends D_i to u_i.

(3) **SecretValueGen:** u_i randomly chooses $\gamma_i \in Z_q^*$ as his/her secret value.

(4) **PublicKeyGen:** u_i uses γ_i to generate the public key $pk_i = g^{\gamma_i}$.

(5) **ShareGen:** Input public keys pk_k (k=1, . . . , n, $k \neq i$) of group users and the secret value γ_i. u_i generates the sharing share as follows.

- u_i selects $\partial_1, \partial_2, \ldots, \partial_{t-1} \in Z_q^*$ and $\iota_1, \iota_2, \ldots, \iota_{t-1} \in Z_q^*$, sets up two polynomials $L(x) = 1/\gamma_i + \partial_1 x + \cdots + \partial_{t-1} x^{t-1}$ and $P(x) = \gamma_i + \iota_1 x + \cdots + \iota_{t-1} x^{t-1}$, where $L(0) = 1/\gamma_i$, $P(0) = \gamma_i$.
- u_i selects $n-1$ random values $x_k \in Z_q^*$, computes $y_k = L(x_k)$, $z_k = P(x_k)$ and $\chi_k = pk_k^{z_k}$.
- u_i saves $F_i = (\chi_1, \chi_2, \ldots, \chi_n)$, and sends (x_k, y_k, z_k) to u_k secretly.

(6) **TagGen:** Input m_j and its name w_j, where $w_j = mid||j||E$. mid is the unique identity information of m_j, j is the index of m_j, and E represents the deleted block or the inserted data. u_i generates the tag by the following steps.

- u_i generates the hash value $N_j = H(w_j)$, the hash value tag $N_j^* = N_j || (N_j)^{\gamma_i}$ and the tag $\sigma_j = D_i \cdot (N_j \cdot g^{m_j})^{\gamma_i}$ for m_j.

- u_i generates $\mathrm{T}_j = w_j||T$, where T denotes the current time stamp.
- u_i generates the authorization $\mathrm{T}_j^* = \mathrm{T}_j||\left(\mathrm{T}_j\right)^{\gamma_i}$.
- u_i sends $\left\{m_j, \sigma_j, F_i\right\}$ and T_j^* to the CSP, and N_j^* to the CP.

(7) **TagVerify:** The CSP and CP first verify the identity of u_i, then the CSP checks the correctness of the tag and the CP stores N_j as follows.

- The CSP verifies the identity of u_i by the following equation.

$$e\left(\left(\mathrm{T}_j\right)^{\gamma_i}, g\right) = e\left(\mathrm{T}_j, pk_i\right).$$

  If it works, the CSP performs the tag verification; otherwise, the CSP rejects the storage request for m_j.
- The CSP checks the correctness of the tag by the following equation.

$$e\left(\sigma_j, g\right) = e\left(H_1\left(ID_i\right), g_1\right) e\left(H\left(w_j\right) \cdot g^{m_j}, pk_i\right).$$

  If the equation holds, the tag is valid. The CSP outputs "1". The CSP stores $\left\{m_j, \sigma_j, F_i\right\}$, signs $\left(\mathrm{T}_j\right)^{\gamma_i}$ with his/her private key ξ, keeps a copy of $\mathrm{T}_j' = \left(\mathrm{T}_j\right)^{\gamma_i \cdot \xi}$, sends T_j' to u_i as a receipt, generates the hash value tag $R_j = H\left(w_j\right)^{\xi}$ and sends R_j to the CP.
  Otherwise, the CSP outputs "0".
- The CP first verifies the identity of u_i by the following equation.

$$e\left(\left(N_j\right)^{\gamma_i}, g\right) = e\left(N_j, pk_i\right).$$

  Then the CP checks the following equation with the public key g^{ξ} of the CSP.

$$e\left(R_j, g\right) = e\left(N_j, g^{\xi}\right).$$

  If they work, the hash value is stored; otherwise, the CP notifies u_i and the CSP of the validation failed.
- After receiving T_j', u_i verifies the validity of the receipt with g^{ξ} as follows.

$$e\left(\mathrm{T}_j', g\right) = e\left(\left(\mathrm{T}_j\right)^{\gamma_i}, g^{\xi}\right).$$

  If the equation holds, u_i deletes local storage $\left\{m_j, \sigma_j\right\}$ and only saves T_j'.

(8) **ChallGen:** Input the set $[1, Z]$ of data index. The challenge is generated as follows.

- u_i sends an auditing request to the TPA.
- The TPA randomly selects a subset C with c elements from the set $[1, Z]$, and chooses c random values $v_j \in Z_q^* \left(j \in C\right)$.
- The TPA sends $chal = \left\{j, v_j\right\}_{j \in C}$ to the CSP.

(9) **ProofGen:** Input $chal$, the CSP generates the proof as follows.

- The CSP selects $\mathcal{A} = \left\{m_j | j \in C\right\}, \mathrm{B} = \left\{\sigma_j | j \in C\right\}$.
- The CSP randomly chooses $\varepsilon \in Z_q^*$, computes $\Gamma = g^{-\varepsilon}$, $\theta = \sum_{j \in C} v_j \cdot m_j$, $\lambda = \varepsilon + \theta$, $\overline{\sigma_j} = \sigma_j^{v_j} \left(j = 1, \ldots C\right)$.
- If σ_j is generated by u_{ij}, the CSP computes $\overline{ID} = \prod_{j \in C} H_1\left(ID_{ij}\right)^{v_j}$ and $\overline{PK} = \prod_{j \in C} pk_{ij}$.
- The CSP sends $proof = \left\{\Gamma, \lambda, \left\{\overline{\sigma_j}\right\}_{j \in C}, \overline{ID}, \overline{PK}\right\}$ to the TPA.

(10) **ProofVerify:** After receiving $proof$, the TPA checks its correctness as follows.

- The TPA checks the correctness of the proof by the following equation.

$$e\left(\prod_{j \in C} \overline{\sigma_j}, g\right) = e\left(\overline{ID}, g_1\right) e\left(\prod_{j \in C} N_j^{v_j} \cdot g^{\lambda} \cdot \Gamma, \overline{PK}\right).$$

  If the following equation holds, the integrity of the data is not destroyed. The TPA outputs "1".
  Otherwise, the data might be tampered or lost. The TPA outputs "0".
- $\left\{N_j\right\}_{j \in C}$ are provided by the CP.

B. SUPPORT USER IDENTITY TRACING

For the malicious user in the group, input t valid group users' sharing share and the information of the destroyed data, the t users could cooperate to track the real identity of the malicious user. The algorithm only involves group users, which provides the fairness in the process of tracing. The specific process is as follows.

- The t valid group users compute $\eta_k = \chi_k^{\gamma_k^{-1}}$ by their share χ_k and secret value γ_k.
- The t valid group users compute $f_p\left(x\right) = \prod_{\substack{p=1 \\ p \neq k}}^{t} \frac{x - x_p}{x_k - x_p}$.
- The t valid group users compute $pk_i = \prod_{k=1}^{t} \eta_k^{f_p(0)}$, which is the public key of the malicious user.

This process ensures that the current malicious group user could be traced. If group users want to trace the previous group user who changes the shared data, they could track the change of shared data by making the postorder-traversal of the binary tree. The group users could find the user identity who has affected data through the above process.

C. SUPPORT USER REVOCATION

When the user u_a revokes from the group, the identity of u_a and all keys of u_a must be immediately declared invalid. Meanwhile, all tags generated by u_a need to be converted to the tags of an existing group user. The algorithm mainly involves five entities, include the revoked user u_a, the existing group user u_b, the other group users u_k, the CSP and the CP, which is shown in Figure 3. The specific process is as follows.

- Firstly, the new signer u_b for the data of u_a is decided by the voluntary application of group users or according to the order in which the user join the group. The group is jointly managed by all group users so that any group user could become u_b and interact with the CSP by the secure channel. Please note that there are no specific restrictions on the selection of the new signer, as long as the new signer is the existing user in the group. Then the CSP chooses $W \in Z_q^*$ and sends it to u_b. Finally, u_b

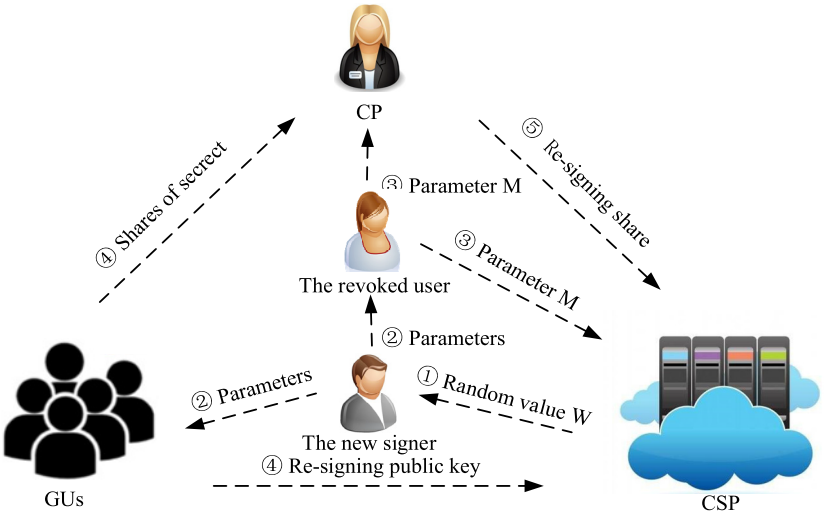

FIGURE 3. User revocation model.

TABLE 2. Private blockchain data structure.

Block identity	Block size	Previous block hash	Visit time	Visitor identity
id	*size*	*hash*	*t*	ID_i

computes D_b^{1/γ_b} and γ_b/W, and sends D_b^{1/γ_b} to u_a, γ_b/W to other group users u_k.

- Given D_b^{1/γ_b} of u_b, u_a computes $M = \frac{\left(D_b^{1/\gamma_b}\right)^{\gamma_a}}{D_a}$. u_k computes the re-signing key share $\mu_k = \gamma_b/W \cdot y_k$ and the re-signing public key $\Upsilon_k = g^{\mu_k}$ utilizing (x_k, y_k) sent by u_a and γ_b/W sent by u_b. u_a sends M to the CP and the CSP. u_k sends (x_k, y_k) to the CP, Υ_k to the CSP.
- After receiving M, (x_k, y_k), m_a and σ_a, the CP checks the correctness of σ_a. If it satisfies the equation $e\left(\sigma_j, g\right) = e\left(H_1\left(ID_i\right), g_1\right) e\left(H\left(w_j\right) \cdot g^{m_j}, pk_i\right)$, the CP computes $\sigma_a^{(k)} = (M \cdot \sigma_a)^{\mu_k}$ and sends the re-signing share $\left(x_k, \sigma_a^{(k)}\right)$ to the CSP.
- After receiving at least t re-signing share $\left(x_k, \sigma_a^{(k)}\right)$, the CSP reorganizes the index of re-signing share into the set K $(k \in K)$. The CSP checks the correctness of $\left(x_k, \sigma_a^{(k)}\right)$ by the equation $e\left(\sigma_a^{(k)}, g\right) = e\left(M \cdot \sigma_a, \Upsilon_k\right)$. If all the re-signing share pass the verification, the CSP computes $\sigma_a' = \prod_{k \in K} \left(\sigma_a^{(k)}\right)^{f_l(0) \cdot W}$.
- u_b updates F_b.

D. SUPPORT DATA DYNAMICS

When group users change, delete, and insert the shared data in the cloud, the data structure based on binary tree could record the change of data. The storage form of data is $\left\{E, m_j^s, \sigma_j^s\right\}$, where s represents modified times of the data. $\left\{m_j^s, \sigma_j^s\right\}$ represents the j th data block has been modified s times. Figure 4 shows the original state of data

FIGURE 4. The original state of data.

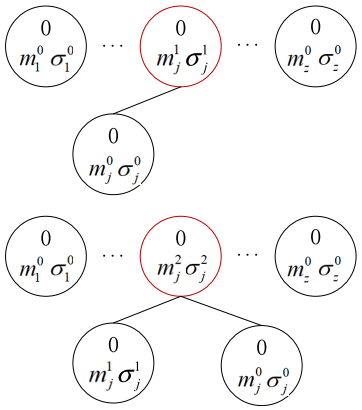

FIGURE 5. The states of data modifications.

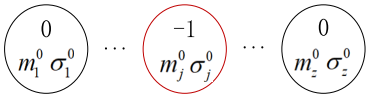

FIGURE 6. The state of data deletion.

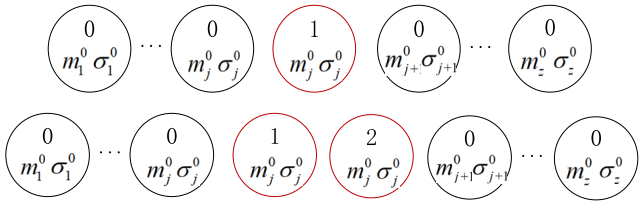

FIGURE 7. The state of data insertions.

$\left\{m_1, \ldots, m_j, \ldots, m_Z\right\}$ and the tags $\left\{\sigma_1, \ldots, \sigma_j, \ldots, \sigma_Z\right\}$. The dynamic data operations are as follows.

(1) **Data Modification:** The data structure of the binary tree could store the last two updated records of the data, and

the latest data would be kept in the root node of the binary tree. Figure 5 shows the state of the data after one modification and two modifications. When the data is lost or damaged, group users could recover the data content by making the postorder-traversal of the binary tree. The structure realizes data traceability and data recoverability.

(2) **Data Deletion:** The record E of the data becomes "-1", as shown in Figure 6.

(3) **Data Insertion:** To avoid disrupting the index of data, the inserted data continues the index of its former data, and adds 1 to E, that is $E+1$. Figure 7 shows the state of the data after one insertion and two insertions.

E. SHARED DATA ACCESSING

When group users want to access the shared data in the cloud, they need to input their identity information and the accessed data identity *mid*. The CSP records the identity ID_i of the data visitor and the specific access time t in the private blockchain, and the data structure of the private blockchain is shown in Table 2. The data visitors could get different degrees of award according to the times of accessing or total access time recorded in the private blockchain. For example, if the evaluation of reward is based on the number of accessing data by data visitors, the CSP could compute the number of access recording blocks in the private blockchain of the data visitor. Otherwise, the CSP could calculate the total access time of the data visitor recorded in the private blockchain. The way encourages users to access and learn shared data more.

V. SECURITY ANALYSIS

In this section, we analyze the correctness and the security requirements of our scheme, and make the security proof.

A. CORRECTNESS ANALYSIS

The correctness of the tag, the proof, identity traceability and the transformed tag in this scheme could be checked by the following derivation.

(1) **The correctness of tags:** The CSP checks the tag of u_i as follows.

$$\begin{aligned} e\left(\sigma_j, g\right) &= e\left(H_1\left(ID_i\right)^{\alpha}\left(H\left(w_j\right)\cdot g^{m_j}\right)^{\gamma_i}, g\right) \\ &= e\left(H_1\left(ID_i\right), g_1\right) e\left(H\left(w_j\right)\cdot g^{m_j}, pk_i\right). \end{aligned}$$

(2) **The correctness of proofs:** The TPA checks the proof sent by the CSP as follows.

$$\begin{aligned} & e\left(\prod\nolimits_{j\in C}\overline{\sigma_j}, g\right) \\ &= e\left(\prod\nolimits_{j\in C}\left(D_{ij}\cdot N_j^{\gamma_i}\right)^{v_j}\cdot g^{\gamma_i\sum_{j\in C}v_j\cdot m_j}, g\right) \\ &= e\left(\prod\nolimits_{j\in C}H_1\left(ID_{ij}\right)^{\alpha\cdot v_j}\cdot\prod\nolimits_{j\in C}N_j^{\gamma_i\cdot v_j}\cdot g^{\gamma_i(\lambda-\varepsilon)}, g\right) \\ &= e\left(\prod\nolimits_{j\in C}H_1\left(ID_{ij}\right)^{v_j}, g_1\right)e\left(\prod\nolimits_{j\in C}N_j^{v_j}\cdot g^{\lambda}\cdot\Gamma, \prod\nolimits_{j\in C}pk_{ij}\right) \\ &= e\left(\overline{ID}, g_1\right) e\left(\prod\nolimits_{j\in C}N_j^{v_j}\cdot g^{\lambda}\cdot\Gamma, \overline{PK}\right). \end{aligned}$$

(3) **The correctness of identity traceability:** The t valid group users compute the public key of the malicious user as follows.

$$\begin{aligned} pk_i &= \prod_{k=1}^{t}\eta_k^{f_p(0)} \\ &= \prod_{k=1}^{t}\chi_k^{\gamma_k^{-1}}\cdot f_p\left(0\right) \\ &= \prod_{k=1}^{t}g^{\gamma_k\cdot z_k\cdot\gamma_k^{-1}}\cdot f_p\left(0\right) \\ &= g^{\sum_{k=1}^{t}z_k\cdot f_p(0)}. \end{aligned}$$

Because of $\sum_{k=1}^{t}z_k\cdot f_p\left(0\right) = P\left(0\right) = \gamma_i$, the public key of the malicious user is $pk_i = g^{\gamma_i}$.

(4) **The correctness of transformed tags:** The tag of the revoked user is transformed into the tag of the existing user as follows.

$$\begin{aligned} \sigma_a' &= \prod_{k\in K}\left(\sigma_a^{(k)}\right)^{f_l(0)\cdot W} \\ &= \prod_{k\in K}\sigma_a^{\mu_k\cdot f_l\left(0\right)\cdot W} \\ &= \prod_{k\in K}\sigma_a^{\gamma_b/W\cdot y_k\cdot f_l\left(0\right)\cdot W} \\ &= \sigma_a^{\gamma_b\sum_{k\in K}y_k\cdot f_l(0)}. \end{aligned}$$

Because of $\sum_{k\in K}y_k\cdot f_l\left(0\right) = L\left(0\right) = 1/\gamma_a$, the re-signing of u_b is $\sigma_a' = \sigma_a^{1/\gamma_a\cdot\gamma_b} = D_b\cdot\left(H\left(w_a\right)\cdot g^{m_a}\right)^{\gamma_b}$.

B. SECURITY REQUIREMENTS

In this subsection, we demonstrate that our scheme satisfies the following security requirements.

(1) **Unforgeability:** The CSP cannot forge any proof without the corresponding data in the process of auditing. Suppose the challenge $chal = \left\{\left\{j, v_j\right\}_{j\in C}\right\}$, the CSP forges the proof $proof = \left\{\Gamma, \lambda', \left\{\overline{\sigma_j}\right\}_{j\in C}, \overline{ID}, \overline{PK}\right\}$. Then we get $e\left(\prod_{j\in C}\overline{\sigma_j}, g\right) = e\left(\overline{ID}, g_1\right)e\left(\prod_{j\in C}N_j^{v_j}\cdot g^{\lambda'}\cdot\Gamma, \overline{PK}\right)$. If the CSP forges the proof successfully, there is an equation $e\left(\sigma_j, g\right) = e\left(H_1\left(ID_i\right), g_1\right)e\left(N_j\cdot g^{m_j}, pk_i\right)$, which is in contradiction with the unforgeability of the signature scheme. That is, if the CSP modifies $\lambda = \varepsilon + \theta$, the valid tags cannot be retrieved.

(2) **Identity privacy:** In the process of auditing, the probability of that the TPA obtains the identities of all signers in the c shared data is about $1/[n\cdot(n-1)\ldots(n-c+1)]$. The probability that anyone obtains the signer's identity of a shared data is about $1/n$. In the process of auditing, due to the randomness of the selected c shared data, the probability that the TPA selects the correct combination of c shared data is $c!/[n\cdot(n-1)\ldots(n-c+1)]$. The total probability that the TPA can distinguish the identities of all the signers from the proof is $1/[n\cdot(n-1)\ldots(n-c+1)]$, which can

be ignored. In the proposed scheme, each shared data is individually signed by one user. However, the TPA cannot distinguish who is the signer of each shared data. Especially in the situations that shared data are modified frequently by different group users. According to the analysis of the scheme, it is proved that the proposed scheme could protect the user identity privacy.

(3) **Data privacy:** The TPA cannot get the corresponding data content from the proof $proof = \left\{\Gamma, \lambda, \left\{\overline{\sigma_j}\right\}_{j\in C}, \overline{ID}, \overline{PK}\right\}$. If the TPA could get $\sum_{j\in C} v_j \cdot m_j$, then the data content can be obtained by collecting numerous linear combinations. Because of $\lambda = \varepsilon + \theta$, in order to solve the bilinear equation, the TPA must get ε from $\Gamma = g^{-\varepsilon}$, which is as difficult as solving the DL problem in G_1. Therefore, the proposed scheme relies on the random masking technology to protect data privacy.

(4) **Collusion resistance:** If the security of the secret sharing technology remains unchanged, the attacker cannot obtain the re-signing key during user revocation. Based on the security of secret sharing technology, the scheme uses Lagrange interpolation polynomial to divide the secret value $1/\gamma_i$ into $n-1$ shares and sends them to other users in the group. The attacker needs to persuade at least $t-1$ users in the group to extract their shares of $1/\gamma_i$ to generate the re-signing key. Considering the cost of this operation, it cannot be realized in practice. Moreover, the scheme introduces the CP in the re-signing process, which makes the collusion attack impossible between the CSP and the revoked user.

C. SECURITY PROOF

In this subsection, we prove that the proposed scheme can resist the attacks of two types of adversaries in the certificateless environment under the random oracle model and the assumption of CDH difficult problem. The specific process is as follows.

Theorem 1: Suppose that there is an attacker $\mathcal{A}_|$ who can win game I with the advantage ϑ_1 that cannot be ignored in the time t_1. If $\mathcal{A}_|$ experiences the most q_{H_1} H_1 hash queries, q_{H_2} H_2 hash queries, q_p partial key queries, q_s secret value queries, q_{pk} public key queries, q_{kr} public key replacement queries and q_t tag queries, there is an algorithm $\mathcal{C}$ with the advantage $\varsigma_1' \geq \varsigma_1/\left(\left(q_p + q_t\right)\cdot 2e\right)$ to solve the CDH problem in time $t_1' \leq t_1 + \left(q_{H_1} + q_{H_2} + q_p + q_s + q_{pk} + q_{kr} + q_t\right)$.

Proof: Suppose the difficult example $\left(G_1, g, g^a, g^b\right)$ of CDH problem, the goal is to calculate g^{ab}.

(1) **System Setup:** $\mathcal{C}$ returns public parameters to $\mathcal{A}_|$ and keeps the system master key ∂ in secret.

(2) **H_1 Hash Query:** $\mathcal{A}_|$ submits the identity ID^* to $\mathcal{C}$, and makes a H_1 hash query. $\mathcal{C}$ maintains the list $L_1 = \{ID, h_1, Q, G\}$, and checks whether $\left(ID^*, h_1^*, Q^*, G^*\right)$ is contained in L_1.

- If $\left(ID^*, h_1^*, Q^*, G^*\right)$ does not exist, $\mathcal{C}$ tosses coins to select $G \in \{0,1\}$. The probability of $G=0$ is τ, the probability of $G=1$ is $1-\tau$. $\mathcal{C}$ randomly chooses $h_1^* \in Z_q^*$. If $G^*=0$, $\mathcal{C}$ calculates $Q^* = g^{h_1^*}$. If $G^*=1$, $\mathcal{C}$ calculates $Q^* = \left(g^b\right)^{h_1^*}$. $\mathcal{C}$ sends Q^* to $\mathcal{A}_|$ and updates L_1.
- If $\left(ID^*, h_1^*, Q^*, G^*\right)$ exists, $\mathcal{C}$ sends Q^* to $\mathcal{A}_|$.

(3) **H_2 Hash Query:** $\mathcal{A}_|$ submits the data identity w^* to $\mathcal{C}$, and makes a H_2 hash query. $\mathcal{C}$ maintain the list $L_2 = \{(w, h_2)\}$. $\mathcal{C}$ looks up w^* in the list L_2.

- If it does not exist, $\mathcal{C}$ randomly selects $h_2^* \in Z_q^*$, sets $g^{h_2^*}$, adds them to L_2, and sends $g^{h_2^*}$ to $\mathcal{A}_|$.
- If it exists, $\mathcal{C}$ sends $g^{h_2^*}$ to $\mathcal{A}_|$.

(4) **Partial Key Query:** $\mathcal{A}_|$ submits the identity ID^* to $\mathcal{C}$, and makes a partial key query. $\mathcal{C}$ maintains the list $L_p = \{(ID, D_{ID}, \gamma_{ID}, pk_{ID})\}$. $\mathcal{C}$ checks whether ID^* and D_{ID^*} are contained in L_p. If ID^* does not exist, $\mathcal{C}$ makes a H_1 hash query. If D_{ID^*} does not exist, $\mathcal{C}$ finds $\left(ID^*, h_1^*, Q^*, G^*\right)$ in L_1 and does these steps as follows.

- If $G^*=1$, $\mathcal{C}$ stops interaction.
- If $G^*=0$, $\mathcal{C}$ sets $D_{ID^*} = (Q^*)^a = g^{ah_1^*}$ and adds D_{ID^*} to the list L_p. $\mathcal{C}$ finds D_{ID^*} in L_p and sends it as the partial key of ID^* to $\mathcal{A}_|$.

(5) **Secret Value Query:** $\mathcal{A}_|$ submits the identity ID^* to $\mathcal{C}$, and makes a secret value query. $\mathcal{C}$ checks whether ID^* and γ_{ID^*} are contained in L_p. If ID^* does not exist, $\mathcal{C}$ makes a H_1 hash query. If γ_{ID^*} does not exist, $\mathcal{C}$ randomly chooses $\gamma_{ID^*} \in Z_q^*$, sets $pk_{ID^*} = g^{\gamma_{ID^*}}$, and adds them to L_p. $\mathcal{C}$ finds γ_{ID^*} in L_p and sends it as the secret value of ID^* to $\mathcal{A}_|$.

(6) **Public Key Query:** $\mathcal{A}_|$ submits the identity ID^* to $\mathcal{C}$, and makes a public key query. $\mathcal{C}$ checks whether ID^* and pk_{ID^*} are contained in L_p. If ID^* does not exist, $\mathcal{C}$ makes a H_1 hash query. If pk_{ID^*} does not exist, $\mathcal{C}$ randomly chooses $\gamma_{ID^*} \in Z_q^*$, sets $pk_{ID^*} = g^{\gamma_{ID^*}}$, and adds them to L_p. $\mathcal{C}$ finds pk_{ID^*} in L_p and sends it as the public key of ID^* to $\mathcal{A}_|$.

(7) **Public Key Replacement Query:** $\mathcal{A}_|$ submits the tuple $\left(ID^*, pk'_{ID^*}\right)$ to $\mathcal{C}$, and makes a public key replacement query. If the tuple $(ID^*, D_{ID^*}, \gamma_{ID^*}, pk_{ID^*})$ does not exist, $\mathcal{C}$ adds $\left(ID^*, pk'_{ID^*}\right)$ to L_p. If the tuple $(ID^*, D_{ID^*}, \gamma_{ID^*}, pk_{ID^*})$ exists, $\mathcal{C}$ replaces the corresponding value in the tuple with $\left(ID^*, pk'_{ID^*}\right)$.

(8) **Tag Query:** $\mathcal{A}_|$ submits (ID^*, w^*, m^*) to $\mathcal{C}$.

- If $G^*=1$, $\mathcal{C}$ stops interaction.
- If $G^*=0$, $\mathcal{C}$ extracts the corresponding $H_2(w^*)$, D_{ID^*} and γ_{ID^*}, calculates the corresponding tag and sends it to $\mathcal{A}_|$.

Forge: $\mathcal{A}_|$ outputs the tag σ' of the data m' on ID' with the public key $pk_{ID'}$.

Analysis: If $\mathcal{A}_|$ wins the game I, $\mathcal{C}$ obtain the equation $e\left(\sigma', g\right) = e\left(H_1\left(ID'\right), g_1\right) e\left(H\left(w'\right)\cdot g^{m'}, pk_{ID'}\right)$.

- If $G^*=0$, $\mathcal{C}$ stops interaction.
- If $G^*=1$, $\mathcal{C}$ sets $H_1\left(ID'\right) = \left(g^b\right)^{h_1'}$, $H_2\left(w'\right) = g^{h_2'}$,

solution of CDH difficult problem by calculating the equation $e\left(\sigma', g\right) = e\left(g^{bh_1'}, g^a\right) e\left(g^{h_2'}\cdot g^{m'}, pk_{ID'}\right)$. The possibility of $g_1 = g^a$. $\mathcal{C}$ could obtain

$g^{ab} = \left(\frac{\sigma'}{(pk_{ID'})^{h_2'+m'}}\right)^{1/h_1'}$ as the challenger $\mathcal{C}$ and adversary $\mathcal{A}_{\text{I}}$ stopping interaction only exists in Partial Key Query and Tag Query, thus the probability that $\mathcal{C}$ outputs g^{ab} is $\varsigma_1' \geq \varsigma_1 \cdot \tau \cdot (1-\tau)^{q_p+q_t} \geq \varsigma_1 / \left((q_p+q_t)\cdot 2e\right)$, the time is $t_1' \leq t_1 + \left(q_{H_1} + q_{H_2} + q_p + q_s + q_{pk} + q_{kr} + q_t\right)$.

Theorem 2: Suppose that there is an attacker $\mathcal{A}_{\text{II}}$ who can win game II with the advantage ϑ_2 that cannot be ignored in the time t_2. If $\mathcal{A}_{\text{I}}$ experiences the most q_{H_1} H_1 hash queries, q_{H_2} H_2 hash queries, q_s secret value queries, q_{pk} public key queries and q_t tag queries, there is an algorithm $\mathcal{C}$ with the advantage $\varsigma_2' \geq \varsigma_2 / ((q_s+q_t)\cdot 2e)$ to solve the CDH problem in time $t_2' \leq t_2 + \left(q_{H_1} + q_{H_2} + q_s + q_{pk} + q_t\right)$.

Proof: Suppose the difficult example $\left(G_1, g, g^a, g^b\right)$ of CDH problem, the goal is to calculate g^{ab}.

(1) **System Setup:** $\mathcal{C}$ returns public parameters to $\mathcal{A}_{\text{II}}$ and keeps the system master key ∂ in secret.

(2) **H_1 Hash Query:** $\mathcal{A}_{\text{II}}$ submits the identity ID^* to $\mathcal{C}$, and makes a H_1 hash query. $\mathcal{C}$ maintains the list $L_1 = \{ID, h_1\}$, and checks whether $\left(ID^*, h_1^*\right)$ is contained in L_1.

- If it does not exist, $\mathcal{C}$ randomly chooses $h_1^* \in Z_q^*$, calculates $g^{h_1^*}$. $\mathcal{C}$ sends $g^{h_1^*}$ to $\mathcal{A}_{\text{II}}$ and updates L_1.
- If it exists, $\mathcal{C}$ sends $g^{h_1^*}$ to $\mathcal{A}_{\text{II}}$.

(3) **H_2 Hash Query:** $\mathcal{A}_{\text{II}}$ submits the data identity w^* to $\mathcal{C}$, and makes a H_2 hash query. $\mathcal{C}$ maintain the list $L_2 = \{(w, h_2)\}$. $\mathcal{C}$ looks up w^* in the list L_2.

- If it does not exist, $\mathcal{C}$ randomly selects $h_2^* \in Z_q^*$, sets $\left(g^b\right)^{h_2^*}$, adds them to L_2, and sends $\left(g^b\right)^{h_2^*}$ to $\mathcal{A}_{\text{II}}$.
- If it exists, $\mathcal{C}$ sends $g^{h_2^*}$ to $\mathcal{A}_{\text{II}}$.

(4) **Secret Value Query:** $\mathcal{A}_{\text{II}}$ submits the identity ID^* to $\mathcal{C}$, and makes a secret value query. $\mathcal{C}$ maintains the list $L_p = \{(ID, \gamma_{ID}, pk_{ID}, G)\}$. $\mathcal{C}$ checks whether ID^* is contained in L_p.

- If it does not exist, $\mathcal{C}$ tosses coins to select $G \in \{0, 1\}$. The probability of $G = 0$ is τ, the probability of $G = 1$ is $1 - \tau$. $\mathcal{C}$ randomly chooses $\gamma_{ID^*} \in Z_q^*$, if $G^* = 0$, sets $pk_{ID^*} = g^{\gamma_{ID^*}}$, adds them to L_p and sends γ_{ID^*} as the secret value of ID^* to $\mathcal{A}_{\text{II}}$; if $G^* = 1$, sets $pk_{ID^*} = (g^a)^{\gamma_{ID^*}}$, adds them to L_p and stops interaction.
- If it exists and $G^* = 0$, $\mathcal{C}$ finds γ_{ID^*} in L_p and sends it as the secret value of ID^* to $\mathcal{A}_{\text{II}}$; if $G^* = 1$, $\mathcal{C}$ stops interaction.

(5) **Public Key Query:** $\mathcal{A}_{\text{II}}$ submits the identity ID^* to $\mathcal{C}$, and makes a public key query. $\mathcal{C}$ checks whether ID^* and pk_{ID^*} are contained in L_p. If ID^* does not exist, $\mathcal{C}$ makes a H_1 hash query. $\mathcal{C}$ checks whether pk_{ID^*} is contained in L_p.

- If it does not exist, $\mathcal{C}$ randomly chooses $\gamma_{ID^*} \in Z_q^*$, if $G^* = 0$, sets $pk_{ID^*} = g^{\gamma_{ID^*}}$, adds them to L_p and sends pk_{ID^*} as the public key of ID^* to $\mathcal{A}_{\text{II}}$; if $G^* = 1$, sets $pk_{ID^*} = (g^a)^{\gamma_{ID^*}}$, adds them to L_p and stops interaction.
- If it exists, $\mathcal{C}$ finds pk_{ID^*} in L_p and sends it as the public key of ID^* to $\mathcal{A}_{\text{II}}$.

(6) **Tag Query:** $\mathcal{A}_{\text{I}}$ submits (ID^*, w^*, m^*) to $\mathcal{C}$.

- If $G^* = 1$, $\mathcal{C}$ stops interaction.
- If $G^* = 0$, $\mathcal{C}$ extracts the corresponding $H_2(w^*)$ and γ_{ID^*}, calculates the corresponding tag and sends it to $\mathcal{A}_{\text{II}}$.

Forge: $\mathcal{A}_{\text{II}}$ outputs the tag σ' of the data m' on ID'.

Analysis: If $\mathcal{A}_{\text{II}}$ wins the game II $\mathcal{C}$ obtain the equation $e\left(\sigma', g\right) = e\left(H_1\left(ID'\right), g_1\right) e\left(H\left(w'\right) \cdot g^{m'}, pk_{ID'}\right)$.

- If $G^* = 0$, $\mathcal{C}$ stops interaction.
- If $G^* = 1$, $\mathcal{C}$ sets $H_1\left(ID'\right) = g^{h_1'}$, $H_2\left(w'\right) = g_2^{bh_1'}$, $pk_{ID'} = g^{a\cdot\gamma_{ID'}}$. $\mathcal{C}$ could obtain $g^{ab} = \left(\frac{\sigma'}{g^{\alpha\cdot h_1' + am'\cdot\gamma_{ID'}}}\right)^{1/h_2'\cdot\gamma_{ID'}}$ as the solution of CDH difficult problem by calculating the equation $e\left(\sigma', g\right) = e\left(g^{h_1'}, g^{\alpha}\right) e\left(g^{bh_2'} \cdot g^{m'}, g^{a\cdot\gamma_{ID'}}\right)$. The possibility of challenger $\mathcal{C}$ and adversary $\mathcal{A}_{\text{I}}$ stopping interaction only exists in Partial Key Query and Tag Query, thus the probability that $\mathcal{C}$ outputs g^{ab} is $\varsigma_2' \geq \varsigma_2 \cdot \tau \cdot (1-\tau)^{q_p+q_t} \geq \varsigma_2 / \left((q_p+q_t)\cdot 2e\right)$, the time is $t_2' \leq t_2 + \left(q_{H_1} + q_{H_2} + q_s + q_{pk} + q_t\right)$.

VI. PERFORMANCE ANALYSIS

In this section, we compare and analyze the proposed scheme and schemes [29], [37] in terms of function implementation, security properties, communication cost, computation cost, and experimental results. For description, we defined the notations used for the section in Table 3. Since the cost of the general hash function operation and pseudo-random number generation operation contributed is negligible, they are not described anymore.

TABLE 3. Notations and descriptions of performance analysis.

Notations	Descriptions
$Mul_{Z_q^*}$, Mul_{G_1}, Mul_{G2}	Multiplication in Z_q^*, G_1, G_2
$Exp_{Z_q^*}$, Exp_{G_1}	Exponentiation in Z_q^*, G_1
$Pair$	Pair operation
n	Number of group users
c	Numbers of challenged data

As we know, an excellent cloud auditing scheme needs to have the characteristics of complete function, safety, low communication cost and low computation cost. In term of function, the cloud auditing scheme needs to satisfy traceability, group user revocation and data dynamic. In term of safety, the cloud auditing scheme needs to satisfy user identity privacy preservation, data content privacy preservation and collusion resistance between the revoked user and the CSP. In terms of communication cost and computation cost, bilinear pairings and exponentiations are very costly computations, which are the main computations for cloud auditing schemes. Meanwhile, tag generation and data auditing are the main phases for cloud auditing schemes. To reduce the communication and computation overhead, schemes need to perform less pair and exponentiation operations in the phases of tag generation and data auditing.

TABLE 4. Comparison of function.

Schemes	Certificateless	Stateless Auditing	Traceability	User Revocation	Non-frameability	Data dynamics	Incentive
Scheme [29]	✓	✗	✗	✗	✗	✗	✗
Scheme [37]	✗	✗	✓	✗	✓	✗	✓
Ours	✓	✓	✓	✓	✓	✓	✓

TABLE 5. Comparison of security.

Schemes	Non-repudiation	Collusion resistance	Data Privacy Preservation	Identity Privacy Preservation
Scheme [29]	✗	✗	✗	✓
Scheme [37]	✗	✗	✓	✓
Ours	✓	✓	✓	✓

TABLE 6. Comparison of communication cost.

Schemes	Challenge	Proof	Total
Scheme [29]	$2c\|q\|+2n\|G\|$	$2\|q\|$	$2(c+1)\|q\|+2n\|G\|$
Scheme [37]	$2c\|q\|$	$(c+1)\|q\|+(n+1)\|G\|$	$(3c+1)\|q\|+(n+1)\|G\|$
Ours	$2c\|q\|$	$\|q\|+(2c+3)\|G\|$	$(2c+1)\|q\|+(2c+3)\|G\|$

A. FUNCTIONALITY COMPARISON

Table 4 shows the comparison of the functional features between the proposed scheme and the related schemes [29], [37]. The scheme [29] is a certificateless cloud auditing scheme with privacy preservation. The scheme [37] is a privacy-preserving cloud auditing scheme for group shared data. Schemes [29], [37] do not have three important properties of stateless auditing, user revocation, and data dynamics. In addition, our scheme also realizes secure data sharing and efficient incentives for data visitors. The scheme in this paper has comprehensive functions, which makes it have more wider application value than the scheme [29] and the scheme [37].

B. SECURITY COMPARISON

Table 5 shows the comparison of the security properties between the proposed scheme and related schemes [29], [37]. We could find that our scheme achieves the non-repudiation between the group user and the CSP, user identity privacy preservation, data content privacy preservation, and could resist the collusion attacks between the revoked user and the CSP. Besides, the proposed scheme does not involve key escrow problems and certificate management problems.

C. NUMERICAL ANALYSIS

We analyze the proposed scheme and previous schemes from communication cost and computation cost in detail.

1) COMMUNICATION COST

The proposed scheme's communication cost in this paper mainly comes from the challenge and the proof in the process of auditing. In the challenge generation stage, the TPA sends the challenge $chal = \{j, v_j\}_{j\in C}$ to the CSP, its cost is $2c\,|q|$, where $|q|$ is the length of Z_q^*. In the proof generation stage, the CSP returns the proof $proof = \left\{\Gamma, \lambda, \{\overline{\sigma_j}\}_{j\in C}, \overline{ID}, \overline{PK}\right\}$ to the TPA, its cost is $|q| + (c + 3)\,|G|$; the CP sends N_j to the TPA, its cost is $c\,|G|$, where $|G|$ is the length of G_1. Thus, the total communication cost of the challenge and the proof in the process of auditing is $(2c + 1)\,|q| + (2c + 3)\,|G|$. To realize stateless auditing, the CP sends the hash value N_j to the TPA to verify the shared data integrity, which produces an extra overhead of $c\,|G|$. Since c is smaller than n, the extra overhead of $c\,|G|$ is negligible. It can be seen from the results that the total communication cost of the proposed scheme is lower than schemes [29], [37], which is shown in Table 6.

2) COMPUTATION COST

The proposed scheme's computation cost in this paper mainly comes from data signing and data auditing. We compare our scheme with schemes [29], [37] on these two stages, and the specific result is shown in Table 7. In the data signing stage, compared with the scheme [29], the proposed scheme has the lower computation cost. On the contrary, the scheme [37] costs lots of computation overhead in the process of signing

TABLE 7. Comparison of computation cost.

Schemes	Tag Generation Phase	Data Auditing Phase
Scheme [29]	$3Mul_{G_1} + 3Exp_{G_1}$	$4Pair + (c+1)Mul_{Z_q^*} + (2c-1)Mul_{G_1} + (2n+3c+3)Exp_{G_1}$
Scheme [37]	$(2n\text{-}1)Mul_{G_1} + 3nExp_{G_1}$	$(n+2)Pair + cMul_{Z_q^*} + (c+1)Exp_{Z_q^*} + (nc-n+c-1)Mul_{G_1} + (nc+n+1)Exp_{G_1} + nMul_{G_2}$
Ours	$2Mul_{G_1} + 2Exp_{G_1}$	$3Pair + cMul_{Z_q^*} + 2(2c-1)Mul_{G_1} + (3c+2)Exp_{G_1}$

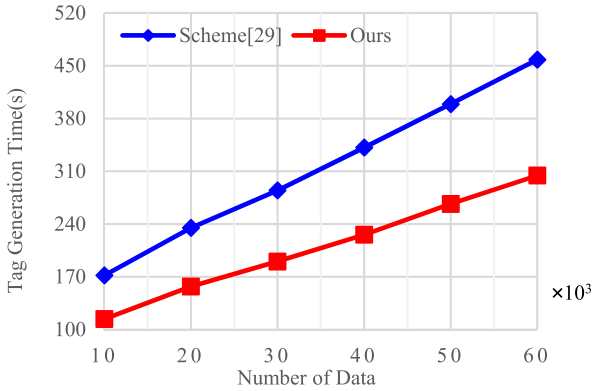

FIGURE 8. Tag Generation phase time.

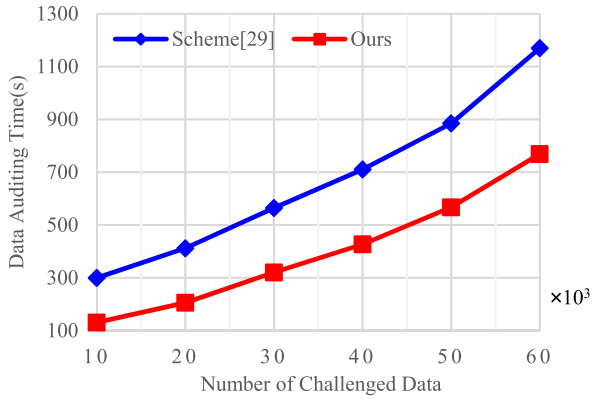

FIGURE 9. Data auditing phase time.

data. The reason is that the scheme [37] utilizes the group signature technology to protect user identity privacy, that is, constructing a complete tag requires adding the public keys of all group users. In the auditing stage, the CSP first computes the proof $\Gamma = g^{-\varepsilon}$, $\lambda = \varepsilon + \theta$, $\overline{\sigma_j} = \sigma_j^{v_j}$ $(j \in C)$, $\overline{ID} = \prod_{j\in C} H_1(ID_{ij})^{v_j}$, $\overline{PK} = \prod_{j\in C} pk_{ij}$, $\theta = \sum_{j\in C} v_j \cdot m_j$. The total computation cost of the proof generation is $cMul_{Z_q^*} + 2(c-1)Mul_{G_1} + (2c+1)Exp_{G_1}$. Then the CSP sends the proof to the TPA. After getting the proof from the CSP, the TPA checks the proof correctness, which the computation cost is $3Pair + 2cMul_{G_1} + (c+1)Exp_{G_1}$. Since our scheme performs the least pairing operation, it is more efficient than schemes [29], [37]. At last, the total computation cost $3Pair + cMul_{Z_q^*} + 2(2c-1)Mul_{G_1} + (3c+2)Exp_{G_1}$ is also the lowest.

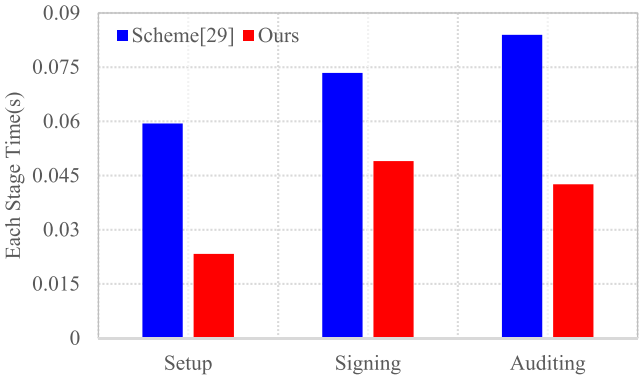

FIGURE 10. Comparison of time in each stage.

D. EXPERIMENTAL RESULTS

In order to better evaluate the proposed scheme in this paper, we compare the proposed scheme with schemes [29], [37] by a series of experiments. These experiments are based on Pairing Based Cryptography (PBC) library and applied to a Windows 10 (64-bit) operating system with an Intel Core i7 3GHz processor with 8 GB RAM. All results are averages of 10 trials.

In the first experiment, we compare the computation overhead of the tag generation in our scheme and the similar scheme [29] by selecting 10000-60000 data. Figure 8 shows the time taken to perform tag generation operation by 5 group users in our scheme and the scheme [29]. We could find that our scheme saves much computational overhead which only spends about 304 seconds to generate tags for 60000 data, and the cost savings increase as the amount of the shared data.

Then we compare the time cost in the proposed scheme with scheme [29] on the different challenged data number and the same user number. As Figure 9 shows, it is clear to see that the more data challenged, the more time overhead it takes to audit, and the TPA has a higher probability of finding problems of the shared data. Although the time of the proposed scheme has a linear relationship with the number of challenged data, our scheme only spends 769s to audit 60000 challenge data, which could be accepted. Besides, compared with the scheme [29], the time growth rate of our scheme is the slowest, which makes the proposed scheme have a massive advantage with the increase of the number of challenged data. Therefore, we could draw a conclusion that our scheme is more efficient during the data auditing process.

We compare the time cost of the proposed scheme and the scheme [29] in the main stages. As Figure 10 shows, our scheme has less time overhead in the stages of Setup, Signing, and Auditing.

VII. CONCLUSIONS

In this paper, we utilize the certificateless signature technology to present a stateless cloud auditing scheme for non-manager dynamic group data with privacy preservation, which avoids the disadvantages of PKC and IBC. During the shared data auditing process, our scheme could satisfy the user identity privacy preservation and data content privacy preservation. Meanwhile, multiple valid group users could trace malicious users' real identities cooperatively, which guarantees security and non-frameability of the scheme. Moreover, the proposed scheme also supports efficient and collusion-resistant user revocation. In order to support data dynamics, we design a data structure based on the binary tree that could support group users to recover their latest data in case of shared data corruption. Furthermore, our scheme assures group users and the TPA stateless, which reduces computation costs in the process of auditing. Our scheme also realizes efficient incentives for data visitors based on the technology of blockchain. In terms of security, the proposed scheme supports secure data sharing and the efficient mutual supervision between group users and the CSP. At last, we show that the proposed scheme could support efficient cloud auditing while resisting two types of attacks in the certificateless environment without jeopardizing the security of shared data in the cloud. In the future, we will extend our scheme to include batch auditing, which could make the TPA fulfill different auditing tasks from multiple users.

REFERENCES

[1] H. Tian, Y. Chen, H. Jiang, Y. Huang, F. Nan, and Y. Chen, "Public auditing for trusted cloud storage services," *IEEE Secur. Privacy*, vol. 17, no. 1, pp. 10–22, Jan. 2019.
[2] C. Cachin, I. Keidar, and A. Shraer, "Trusting the cloud," *ACM SIGACT News*, vol. 40, no. 2, pp. 81–86, Jun. 2009.
[3] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Comput.*, vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.
[4] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 8, pp. 1034–1038, Aug. 2008.
[5] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, May 2011.
[6] L. Chen, S. Zhou, X. Huang, and L. Xu, "Data dynamics for remote data possession checking in cloud storage," *Comput. Electr. Eng.*, vol. 39, no. 7, pp. 2413–2424, Oct. 2013.
[7] S. Yu, "Big privacy: Challenges and opportunities of privacy study in the age of big data," *IEEE Access*, vol. 4, pp. 2751–2763, 2016.
[8] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 9, pp. 1432–1437, Sep. 2011.
[9] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 362–375, Feb. 2013.
[10] Y. Yu, H. A. Man, Y. Mu, S. Tang, and J. Ren, "Enhanced privacy of a remote data integrity-checking protocol for secure cloud storage," *Int. J. Inf. Secur.*, vol. 14, no. 4, pp. 307–318, 2015.
[11] J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Trans. Services Comput.*, early access, Jan. 8, 2018, doi: 10.1109/TSC.2018.2789893.
[12] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," in *Proc. ACNS*, Berlin, Germany, 2012, pp. 507–525.
[13] B. Wang, H. Li, and M. Li, "Privacy-preserving public auditing for shared cloud data supporting group dynamics," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Budapest, Hungary, Jun. 2013, pp. 1946–1950.
[14] B. Wang, B. Li, and H. Li, "Panda: Public auditing for shared data with efficient user revocation in the cloud," *IEEE Trans. Services Comput.*, vol. 8, no. 1, pp. 92–106, Jan. 2015.
[15] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 43–56, Jan. 2014.
[16] C. Liu, J. Chen, L. T. Yang, X. Zhang, C. Yang, R. Ranjan, and K. Rao, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2234–2244, Sep. 2014.
[17] Y. Yu, J. Ni, M. H. Au, Y. Mu, B. Wang, and H. Li, "Comments on a public auditing mechanism for shared cloud data service," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 998–999, Nov. 2015.
[18] Y. Zhang, C. Xu, X. Liang, H. Li, Y. Mu, and X. Zhang, "Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 3, pp. 676–688, Mar. 2017.
[19] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, Alexandria, VA, USA, 2007, pp. 598–609.
[20] A. Juels and B. S. Kaliski, "PORs: Proofs of retrievability for large files," in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, Alexandria, VA, USA, 2007, pp. 584–597.
[21] Z. Xu, L. Wu, M. K. Khan, K.-K.-R. Choo, and D. He, "A secure and efficient public auditing scheme using RSA algorithm for cloud storage," *J. Supercomput.*, vol. 73, no. 12, pp. 5285–5309, Dec. 2017.
[22] H. Wang, "Identity-based distributed provable data possession in multicloud storage," *IEEE Trans. Services Comput.*, vol. 8, no. 2, pp. 328–340, Mar. 2015.
[23] X. Yang and Y. Li, "Revocable identity-based proxy re-signature scheme in standard model," *J. Commun.*, vol. 40, no. 5, pp. 153–162, 2019.
[24] Y. Yu, L. Xue, M. H. Au, W. Susilo, J. Ni, Y. Zhang, A. V. Vasilakos, and J. Shen, "Cloud data integrity checking with an identity-based auditing mechanism from RSA," *Future Gener. Comput. Syst.*, vol. 62, pp. 85–91, Sep. 2016.
[25] H. Wang, D. He, and S. Tang, "Identity-based proxy-oriented data uploading and remote data integrity checking in public cloud," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 6, pp. 1165–1176, Jun. 2016.
[26] B. Wang, B. Li, H. Li, and F. Li, "Certificateless public auditing for data integrity in the cloud," in *Proc. CNS*, Washington, DC, USA, Oct. 2013, pp. 136–144.
[27] J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Trans. Services Comput.*, early access, Jan. 8, 2018, doi: 10.1109/TSC.2018.2789893.
[28] K. He, C. Huang, K. Yang, and J. Shi, "Identity-preserving public auditing for shared cloud data," in *Proc. IEEE 23rd Int. Symp. Qual. Service (IWQoS)*, Portland, OR, USA, Jun. 2015, pp. 159–164.
[29] G. Wu, Y. Mu, W. Susilo, F. Guo, and F. Zhang, "Privacy-preserving certificateless cloud auditing with multiple users," *Wireless Pers. Commun.*, vol. 106, no. 3, pp. 1161–1182, 2019.
[30] A. Fu, S. Yu, Y. Zhang, H. Wang, and C. Huang, "NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users," *IEEE Trans. Big Data*, early access, May 5, 2017, doi: 10.1109/TBDATA.2017.2701347.
[31] T. Yang, B. Yu, H. Wang, J. Li, and Z. Lv, "Cryptanalysis and improvement of Panda-public auditing for shared data in cloud and Internet of Things," *Multimedia Tools Appl.*, vol. 76, no. 19, pp. 19411–19428, 2017.
[32] Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, and G. Min, "Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 767–778, Apr. 2017.
[33] Y. Zhu, G. J. Ahn, H. Hu, S. S. Yau, H. G. An, and S. Chen, "Dynamic audit services for outsourced storages in clouds," *IEEE Trans. Services Comput.*, vol. 6, no. 2, pp. 227–238, Apr./Jun. 2011.

[34] H. Zhao, X. Yao, X. Zheng, T. Qiu, and H. Ning, "User stateless privacy-preserving TPA auditing scheme for cloud storage," *J. Netw. Comput. Appl.*, vol. 129, no. 1, pp. 62–70, Mar. 2019.
[35] N. Garg and S. Bawa, "RITS-MHT: Relative indexed and time stamped merkle hash tree based data auditing protocol for cloud computing," *J. Netw. Comput. Appl.*, vol. 84, pp. 1–13, Apr. 2017.
[36] Z. Mo, Y. Zhou, S. Chen, and C. Xu, "Enabling non-repudiable data possession verification in cloud storage systems," in *Proc. IEEE 7th Int. Conf. Cloud Comput.*, Anchorage, AK, USA, Jun. 2014, pp. 232–239.
[37] L. Huang, G. Zhang, and A. Fu, "Privacy-preserving public auditing for non-manager group shared data," *Wireless Pers. Commun.*, vol. 100, no. 4, pp. 1277–1294, Jun. 2018.
[38] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

XIAODONG YANG (Member, IEEE) received the B.S. degree in mathematics from Northwest Normal University and the M.S. degree in cryptography from Tongji University, China, in 2002 and 2005, respectively, and the Ph.D. degree in cryptography from Northwest Normal University, in 2010. He is currently a Postdoctoral Fellow with the State Key Laboratory of Cryptology of China and a Professor in Information and Computer Science with Northwest Normal University. His research interests include applied cryptography, network security, and cloud computing security. He is also a member of the Chinese Cryptology and Information Security Association.

MEIDING WANG received the B.S. degree from Northwest Normal University, Lanzhou, China, in 2018, where she is currently pursuing the master's degree in computer science. Her current research interest includes cloud computing security.

XIUXIU WANG received the B.S. degree from Northwest Normal University, Lanzhou, China, in 2019, where she is currently pursuing the master's degree. Her current research interest includes digital signature.

GUILAN CHEN received the B.S. degree from Northwest Normal University, Lanzhou, China, in 2018, where she is currently pursuing the master's degree in computer science. Her current research interest includes searchable encryption.

CAIFEN WANG received the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2003. She is currently a Professor in Computer Science with Shenzhen Technology University. Her current research interests include network security, cryptographic protocols, and security engineering. She is also a member of the Chinese Cryptology and Information Security Association.