IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 23, 2019, accepted July 29, 2019, date of publication August 13, 2019, date of current version August 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2935180

Shared Dynamic Data Audit Supporting Anonymous User Revocation in Cloud Storage

YINGHUI ZHANG[1,2] (Member, IEEE), CHEN CHEN[1], DONG ZHENG[1,2], RUI GUO[1], AND SHENGMIN XU[3]

[1]National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, Xi'an 710121, China
[2]Westone Cryptologic Research Center, Beijing 100070, China
[3]School of Information Systems, Singapore Management University, Singapore

Corresponding authors: Yinghui Zhang (yhzhaang@163.com) and Dong Zheng (zhengdong@xupt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802000, in part by the National Natural Science Foundation of China under Grant 61772418, Grant 61472472, and Grant 61802303, in part by the Key Research and Development Program of Shaanxi under Grant 2019KW-053, in part by the Natural Science Basic Research Program of Shaanxi under Grant 2018JZ6001, and in part by the Sichuan Science and Technology Program under Grant 2017GZDZX0002. The work of Y. Zhang was supported by the New Star Team of Xi'an University of Posts and Telecommunications under Grant 2016-02.

ABSTRACT Collusion between revoked users and cloud service providers can pose a threat to the security of cloud storage data. If the original legitimate users cannot be revoked securely, it will lead to the leakage of shared data, thus affecting the security of cloud storage. In this paper, we combine vector commitment and anonymous revocation of group signature to propose an integrity audit scheme for cloud storage data that can support data modification. The anonymity of the group signature ensures that users' privacy information will not be snooped by the server. The proposed scheme supports the dynamic operation of stored data by legitimate group users besides data owners. When the user behaves improperly, the membership can be revoked by the group manager. After the user-modified data is stored in the cloud, whether the cloud server correctly stores the data can be audited by a trusted third party. Security analysis and experimental results demonstrate that our scheme is secure and efficient.

INDEX TERMS Cloud storage, group signature, integrity audit, user revocation, vector commitment.

I. INTRODUCTION

With the development and improvement of cloud computing technology, many individuals and enterprises outsource data to cloud service provider (CSP) for data computing and data storage. While cloud storage brings many conveniences to our lives and technological developments, it also faces various security threats. When the user delegates the cloud service provider to store the data. If there is no supervision mechanism for third-party platforms, users'data may be maliciously tampered with or deleted by CSP. To solve this problem, an integrity auditing scheme for cloud storage data has been proposed. In the initial audit scheme, in order to verify the integrity of the data, users need to calculate and save the corresponding hash value of the data before uploading the data to the cloud server (CS). By comparing the hash values, the user can determine whether the data stored on the cloud server is corrupted. However, as the number of data increases. Each time traversing the entire data for calculation increases the computational complexity, reduces audit efficiency, and greatly increases communication costs. This approach is not practical for users with limited computing and storage capabilities.

Auditing schemes for cloud storage data can be categorized as audits for dynamic data [1]–[3], semi-dynamic data [4], and static data. Most of the research now focuses on auditing data that can support dynamic modifications. In terms of data modification permissions, the original scenario only supports data owners to dynamically manipulate data. However, as the demand for shared data increases, many auditing schemes are proposed to support group users to modify data [5], [6]. There are many audit schemes only consider how to verify the integrity of data and the correctness of data storage. However, there is no corresponding protection

The associate editor coordinating the review of this article and approving it for publication was Shagufta Henna.

for users' identity privacy when they sign data blocks. If the privacy protection of users' data is not taken into account, users' data will face the risk of leakage, which will cause security problems that can not be ignored in cloud storage and hinder the development of cloud computing. In 2010, Wang *et al*. proposed a scheme [7] using the random mask and the public key based homomorphic authenticator technique to protect data privacy but does not support users to modify the data. To protect the shared data from tampering and deletion, [8] introduced a trusted third-party (TPA) to audit the data. It uses the idea of proxy re-signing to implement an effective user revocation mechanism. In addition to data audit in cloud computing, there are also many other relevant researches including fine-grained access control [9], encrypted data search [10], identity-based authentication [11] and data crowdsensing [12].

In the integrity auditing scheme of cloud storage data, implementing secure user revocation ensures that data is shared among legitimate group users. More specifically, if the revoked user's access to data is not managed, then the CSP and revoked user will collude for profit reasons, resulting in the corruption of data. Users in a group share the group private key to generate signatures. When a user is revoked, the group manager (GM) updates the group private key without distributing the new private key to the user who needs to be revoked, and the remaining users update their signatures according to the new group private key. But this approach can bring huge overhead to communication and computing. Because of the process of generating a signature, the user needs to generate a signature again based on the messages stored in the cloud. In this paper, we use the revocation list (RL) to manage the user's revocation, the *tag* is part of the signature, and the revoked user's *tag* is stored in the RL. When the user is revoked, the ability of the remaining users will not be affected. Since the *tag* generated by the revoked user is invalid, the cloud server can reject the user access or update of the shared data after verifying the signature containing the invalid *tag*.

A. OUR CONTRIBUTION

Data stored on cloud servers is shared among legitimate group users. The shared data supports the user to modify it and other update operations. In order to prevent the collusion between revoked group users and CSP from leaking and tampering with shared data, we need to provide a secure user revocation mechanism while implementing efficient data audit. So we propose a shared dynamic data audit scheme with anonymous revocation of users. The contributions are summarized as follows:

- Based on the dynamic user group, users in the group can be safely revoked.
- The proposed scheme supports the ciphertext database being shared among multiple users, and the user has the right to modify data operations. It also provides efficient data integrity auditing.
- Based on comprehensive security analysis and experimental results, we show that the proposed scheme is secure and efficient.

B. ORGANIZATION

The content of this article is organized as follows: Section II describes our system/threat model and design goals. Section III introduces some preliminaries involved in this paper. Section IV describes the specific construction of the scheme, and in Section V analyzes the correctness and security of the program. Section VI is an analysis of our experimental results. The introduction of related work is reviewed in Section VII. The work of the entire article is summarized in Section VIII.

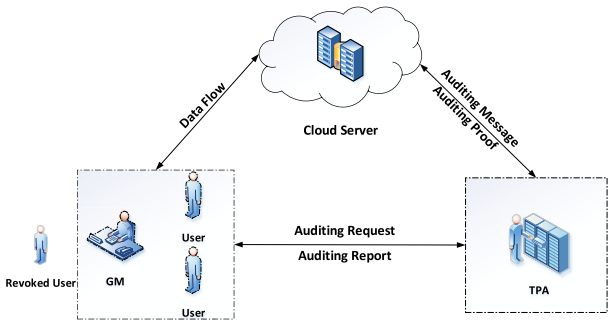

FIGURE 1. The system model.

II. PROBLEM STATEMENT

A. SYSTEM MODEL

The system model of our scheme consists of a cloud service provider and the corresponding cloud service consumer referred to herein as a valid group user and a trusted third-party auditor. As shown in Figure 1, Group users can be divided into group members and GM. The group manager is responsible for undoing group users and adding new group users. Group users outsource data to cloud servers, and only users who have registered with the GM can access or modify the shared data stored by the group on the cloud. When users attempt to access or modify data, CSP verify the identity of users. Decide whether to provide data services to users according to their legitimacy. However, CSP is not completely trusted, so TPA is used to detect CSP misconduct.

Stored data is divided into many data blocks, which constitute a sequence of messages. Vector commitment scheme is used to commit to message sequence and generate proof of commitment. And users can update messages in different positions of message sequence and generate corresponding proofs. After the user updates the message at a certain location, the third-party auditor also generates an updated certificate about the vector commitment of the updated message sequence according to the update information to verify the validity of the update.

B. THREAT MODEL

The threats we consider are mainly from the following three aspects:

- Threat from semi-trusted CSP: CSP may conceal data corruption by dishonestly reporting data storage status to users.
- Threats from revoked users: Revoked users collude with CSP to make non-conforming modifications to data.
- Storage of plaintext data may lead to the disclosure of some private information about data.

C. DESIGN GOALS

The design goals are described below:

- **Public Audit:** TPA does not need to traverse the entire data when auditing according to the user's requirements. Using the advantages of TPA can reduce the audit burden of users.
- **Correctness:** Verification ensures that data is stored correctly and its integrity is not compromised.
- **Secure user revocation:** After the user is revoked, the corresponding signature expires and the legal membership is lost, and the user can no longer access or sign the shared data.
- **Dynamic update:** While supporting the dynamic operation of data by users, it can verify whether the operation of corresponding data is correctly executed.
- **Traceability:** Signatures generated by legitimate users can be tracked by GM so that malicious group users can be revoked in time when malicious behavior occurs.

III. PRELIMINARIES AND COMPLEXITY ASSUMPTIONS

A. BILINEAR GROUPS

Let G, G_T are two multiplicative cyclic groups of prime order p. e is a bilinear map: $G \times G \rightarrow G_T$ with the following properties:

- Bilinearity: For all $u, v \in G$ and $a, b \in Z_p$, $e(u^x, v^y) = e(u, v)^{xy}$;
- Non-degeneracy: $e(g, g) \neq 1$;
- Computability: there is an effective algorithm to compute bilinear maps e.

B. COMPLEXITY ASSUMPTION

The security of this scheme can be obtained through the difficulties of the following problems.

- Square Computational Diffie-Hellman problem (SCDH): Let G be a cyclic group of prime order q. Suppose g is the generator of G. For all $u, v \in Z_p$ on input g, g^u to computing g^{u^2} as output.
  In this paper, we construct vector commitment based on CDH assumption in bilinear groups. Its security is equivalent to the SCDH assumption.
- q-strong Diffie-Hellman problem (q-SDH): Let G, G_T defines as above, where possibly $G = G_T$. And let g_1 be a generator of G and g_2 be a generator of G_T. Given a $(q+2)$-*tuple*$(g_1, g_2, g_2^u, g_2^{u^2}, \ldots, g_2^{u^q})$ as input, output a pair $(g_1^{1/u+v}, v)$ where $v \in Z_p$.
  This assumption is used in this paper to construct short group signatures to ensure that the user's signature is unforgeable.
- Decision Linear problem (DLIN): g_1 is a generator of G. Randomly select generators $u, v, w \in G$. Given $u, v, w, u^a, v^b, w^c \in G$ as input. Output 1 if $a + b = c$ and no otherwise, where $u, v, w \in_R G$ and $a, b, c \in_R Z_p$. DLIN problems are difficult to solve even in the bilinear group G.

C. VECTOR COMMITMENT

Vector commitment has two important attributes. One of the properties is *hiding* and the other is *binding*. The properties of *hiding* requires commitment should not reveal information about the message, which has been committed. And the properties of *binding* requires the user who make a commitment to messaging cannot change the information of the message. More precisely, commitments with an effective open verification process, the verifier can effectively verify an open message to the original one.

Vector Commitment is proposed by Catalano and Fiore [13] which is an extension of the commitment primitives. Unlike the standard commitment scheme, the vector commitment scheme requires a security requirement, namely *position binding*. For any PPT adversary $\mathcal{A}$ wants to check the different values of the same commitment at the same position is infeasible. Meanwhile, the size of commitment C_v is independent of vector length n so is the output of the opening algorithm.

There is $h_i = g^{z_i}$ for each $z_i \in_R Z_p$, in which case the public key for the vector commitment is $pk_{vc} = \{h_1, \ldots, h_n\}$. The message sequence is $(m_1, \ldots, m_n)$ that we consider as a vector. The commitment value C_v is calculated for this vector by the formula $C_v = \prod_{i=1}^{n} h_i^{m_i}$. And in order to facilitate the verification of the correctness of the message, we need to calculate its proof

$$\Lambda_i^t = \prod_{j=1, j\neq i}^{n} h_{i,j}^{m_j} = \Big(\prod_{j=1, j\neq i}^{n} h_j^{m_j} \Big)^{z_i}$$

for each sequence. If the verification equation $e(C_v^t / h_i^{m_i}, h_i) = e(\Lambda_i^t, g)$ is satisfied, we can confirm the correctness of the message sequence.

D. GROUP SIGNATURE

The anonymous revocable short group signature scheme [14] used in this paper consists of four algorithms: KeyGen, Join, Sign, and Verify. The specific steps are as follows:

- *KeyGen*: The GM takes the parameters generated in the **Setup** phase as the input of the *KeyGen* algorithm. Finally, the *KeyGen* algorithm outputs group public key $gpk = (param, g^{k_m})$ and GM's secret key $msk = k_m$.
- *Join*: The input of the algorithm is gpk and k_m. In the process of interaction, the algorithm is assumed by GM and User respectively. GM manages the

user's participation. User calculates its secret key $k_u[i]$ through this algorithm.

- *Sign*: This algorithm takes *gpk* and $k_u[i]$, a RL containing the revoked tag of the revoked user, and the commitment value C_v of the committed message as input. Output signature σ.
- *Verify*: The input of the verification algorithm consists of *gpk*, signature σ on committed message, and RL. The output is 1 or valid, meaning that the signature was signed by a legitimate group of users.

IV. SCHEME CONSTRUCTION

A. A CONCRETE SCHEME

- **Setup:** First, We need to make the following parameter settings. Our scheme uses the bilinear group G, G_T as explained in the previous section. The properties are as described above. And the range of subscripts i, j appearing in the scheme is explained in advance from 1 to n. Choose $z_1, \ldots, z_n \in Z_p$ random. Set: $h_i = g^{z_i}$,and $h_{i,j} = g^{z_i z_j}$. Given the upper limit of outsourced data blocks is n. Set the message space as $\mathcal{M}$, and there is message sequence $m_i \in \mathcal{M}$. 1^λ is provided as input to the algorithm, where λ is called the security parameters. Select hash functions $H_G : \{0,1\}^* \rightarrow G$, and $H : \{0,1\}^* \rightarrow Z_p$. Randomly select element g, g_1, g_2 from G Then output $param = (p, G, G_T, e, H_G, H, g, g_1, g_2, \{h_i\}_{i\in[n]}, \{h_{i,j}\}_{i,j\in[n],i\neq j})$ as public parameters.
- **Signature Key Generation:** Public parameters are used as input to this algorithm. Select $k_m \in Z_p$ and compute $k_v = g^{k_m}$, Where k_m is the secret key of the group manager. Output group public key $gpk = (p, G, T, e, H_G, H, g, g_1, g_2, k_v)$ and secret key k_m.
- **Vector Commitment Generation:** Using the algorithm $\text{VC.Com}_{pp}(m_1, \ldots, m_n)$ to compute vector commitment $C_v = \prod_{i=1}^n h_i^{m_i}$ corresponding to the message sequence and output the description information $des = (m_1, \ldots, m_n)$ and C_v.
- **User Registration:** The i-th user U_i selects $u_i, v_i' \in Z_p$, and computes $S_i = g_1^{u_i} \cdot g_2^{v_i'}$ then sends S_i to GM. GM computes $S_i' = (S_i \cdot g_2^{v_i''} \cdot g)^{1/(k_m+z_i)}$ for $v_i'', z_i \in_R Z_p$, and return (S_i', v_i'', z_i) to U_i. U_i check if the equation $e(S_i', k_v \cdot g^{z_i}) \stackrel{?}{=} e(g_1^{u_i} g_2^{v_i'} g, g)$ is hold by calculating $v_i = v_i' + v_i'' mod\ p$. Under the condition of $S_i'^{k_m+z_i} = g_1^{u_i} \cdot g_2^{v_i} \cdot g$, users can get secret keys $k_u[i] = (S_i', u_i, v_i, z_i)$ that belong only to themselves.
- **Signature Generation:** In this paper, user signatures include user identity authentication commitments C, *tag*, invalid certificates of tags *cert*, and knowledge signatures *SPK* used to confirm the membership relationship between users and other groups of users,etc. In order to ensure the freshness of vector commitment C_v and signature σ^t, t is used as a counter. Through t, we can confirm that the current data and corresponding signatures and vector commitment are up-to-date.
  a. Choose a random number r from Z_p and use the parameters obtained in the above steps to compute $\varphi = H_G(gpk||(C_v(t-1), C_v^t, t)||r)$.
  b. Compute $C = S_i' \cdot g_2^\alpha$, where $\alpha \in_R Z_p$.
  c. Calculate user tags, select a random value $\beta \in_R Z_p$, copmute $\mu = g^{u_i+\beta}$ and $\zeta = \varphi^\beta$. Then get the tag as $tag = (\varphi, \mu, \zeta)$.
  d. For all $1 \leq j \leq k$, select $\delta_j, \theta_j \in_R Z_p$. k represents the number of revoked users. Compute $\Gamma_j = g^{\theta_j(u_i+\delta_j)}, \Phi_j = \varphi_j^{\delta_j}, \Omega_j = g^{\theta_j}, \Upsilon_j = \varphi_j^{\theta_j}$. Then output the invalid cerificates of tag $cert_j = (\Gamma_j, \Phi_j, \Omega_j, \Upsilon_j)$.
  e. The knowledge information $K = (u_i, \theta_1, \ldots, \theta_k, \alpha, z_i, \beta, \varepsilon, \delta_1, \ldots, \delta_k)$ is selected to generate the $\tau = SPK\{(K) : tag_i, cert_i, e(C, k_v)/e(g, g) = e(g_1, g)^{u_i} e(g_2, g)^{\varepsilon} e(g_2, k_v)^{\alpha} / e(C, g)^{z_i}\}$.
  f. Output signature $\sigma^t = (r, C, tag, cert_1, \ldots, cert_k, \tau)$. In reality, cloud servers provider first verifies the validity of signatures. If the signature passes verification, the CSP calculates the commitment $C_v(t) = \sigma^t \cdot (\prod_{i=1}^n h_i^{c_i^t})$ containing the signature and saves the result and the current number of data updates t in the description information *des*.
- **Proof Generation:** Group users use $\text{VC.open}_{pp}(m_i^t, i, des)$ to compute a opening proof

$$\Lambda_i^t = \prod_{j=1, j\neq i}^{n} h_{i,j}^{m_j} = (\prod_{j=1, j\neq i}^{n} h_j^{m_j})^{z_i}$$

of the committed message, which is the *i-th* position in the message sequence. Then ruturn Λ_i^t to verifier. Λ_i is used to ensure that C_v^t is generated by the message sequence during the verification process. Embodies the attributes of the *position binding*. The correctness of commitment value is related to its position. Commitment values vary from position to position. And generate a set of information about the proof $\omega = (m_i, \Lambda_i, C_v, t)$ as a return.
- **Verify:** The verification in this scheme is divided into signature verification and data verification. The specific process is as follows:
  a. Signature verification: With *gpk*, revocation list RL= $(cert_1, \ldots, cert_k)$, signature σ^t and corresponding message $\{C_v^t, t\} \in Z_p^*$ as input for signature verification. First, check τ. Second, for all $1 \leq i \leq k$ verify whether $e(\varphi_j, \Gamma_j)/e(\Phi_j, \Omega_j) \neq e(\Upsilon_j, \mu_j)/e(\zeta_j, \Omega_j)$ is true through the inequality $e(\varphi_j, \Gamma_j) \neq e(\Upsilon_j, \mu_j)/e(\zeta_j, \Omega_j)$. If all verification is established, this signature is a valid signature.
  b. Data Verification: After the verifier receives the proof Λ_i^t and ω, it uses $\text{VC.Ver}(C_v^t, m_i^t, i, \omega)$ to verify whether the equation $e(C_v^t/h_i^{m_i}, h_i) \stackrel{?}{=} e(\Lambda_i^t, g)$ is true. Output 1 if the equation is true. This means that this proof is generated by the message sequence $m_1, \ldots, m_n$. Otherwise output $\perp$.

- **Update:** A group user who produced C_v according to message m and wants to update m to m'. Compute new commitments $C'_v = C_v \cdot h_i^{m-m'}$. The new commitment C'_v and the user signature is sent to the CSP as an update request. After receiving the request, the CSP verifies the signature. If ture, the CSP updates the data and stores the corresponding committed value.
- **Dynamic data operation:** The dynamic operation of the cloud storage data by the users involved in this paper mainly includes three aspects: The insertion of data at different locations,that is, the addition of data; The modification of data; The deletion of data.
  a. Data addition:
  Suppose the user wants to insert data m at the i-th position after the x-th data block m_x. The data block index after the $x+i$ position changes due to the insertion of new data blocks. Users need to calculate the new commitment value $C'_v = C_v \cdot h_i^{m_x-m}$ based on the new message sequence for the generation of new signatures σ'. Users send data insertion requests $R_{insert} = \{x+i, m, \sigma'\}$ to CSP. After receiving the request, the cloud server first verifies whether the signature is from a valid group user through $e(\varphi_j, \Gamma_j)/e(\Phi_j, \Omega_j) \neq e(\Upsilon_j, \mu_j)/e(\zeta_j, \Omega_j)$. When the verification passes, the original data m_x is replaced with the new data m and inserted into the user-specified position. Finally, the cloud server calculates the committed value of the new message sequence and stores it, and the correctness of the storage can be verified by $e(C_v^t/h_j^m, h_j) = e(\Lambda_j^t, g)$.
  b. Data modification:
  When the user wants to modify the x-th data m_x in the message sequence. First, the user calculates the new commitment values and signatures for the message sequence, which are C'_v and σ', respectively. Secondly, the update request $R_{update} = \{x, m'_x, \sigma'\}$ containing the update data position information x is sent to the CSP. Finally, the CSP updates the corresponding position data m_x to m'_x. The verification of signature and storage correctness is similar to the addition of data.
  c. Data deletion:
  After the user specifies that the position x of the data block m_x is to be deleted, the data deletion request $R_{delete} = \{x, \sigma'\}$ is sent to the CSP. The CSP first verifies the user's signature, confirms that it is a legitimate user, deletes the data of the user-specified position according to the data deletion request, and updates the signature of the data block to σ'.
- **ProofUpdate:** The updated proof Λ_j can be calculated from the new commitment C_v. Assume that the updated message m_i has proof Λ_j. Because vector commitment has the security requirement of *position binding*, so we need to distinguish between two different situations:
  a. $i \neq j$, compute the update proof $\Lambda'_j = \Lambda_j \cdot (h_i^{m'-m})^{z_j} = \Lambda_j \cdot h_{i,j}^{m'-m}$.
  b. $i = j$, compute the update commitment as $C'_v = C_v \cdot h_i^{m'-m}$ and remain the same proof $\Lambda_j = \Lambda_i$.
- **User Anonymous Revocation:** Anonymous revocation of users is performed by the GM. When the GM receives the signature, GM first verifies whether the *tag* belongs to valid group user by verifying $tag \stackrel{?}{=} (H(gpk||C_v(t-1), C_v^t, t||r), \mu, \zeta)$. If the user's identity is determined to be legitimate, GM compares the user *tag* with the *tag* stored in *RL*. Since the tag is generated by using the $k_u[i]$. And the user secret key is derived from the interaction with the GM, so the tag is unique. If the user *tag* does not match, the user has not been revoked.

B. DATABASE ENCRYPTION

When the amount of local data exceeds the user's load capacity, the user outsources the data to a server that can store the data. Under current technology development, outsourcing services in the cloud storage environment is the best choice for users. To protect the security of data, users want to have their database encrypted before uploading to the cloud. Our scheme supports data modification, deletion and other data update operations. However, simply using a symmetric encryption scheme does not support multiple user operations scenarios. In the same group, group users share encrypted data. To conveniently decrypt the data, a decryption key is shared among the group users. However, when the user is revoked, the key may be leaked to pose a threat to data security. Therefore, we need to adopt appropriate encryption schemes to maintain the data security of legitimate users during the revocation of revoked users. If the revoked user intentionally reveals the decryption key, data security is compromised. Encryption schemes should ensure that malicious attackers cannot compute the decryption keys of other users.

To overcome the problem of malicious users leaking shared keys. Jiang *et al.* [5] used the asymmetric group key agreement (ASGKA) protocol to negotiate shared keys. The confidentiality of data is achieved while ensuring key security. However, in a round of ASGKA protocol, the number of group members will affect the length of ciphertext, and a large number of users will also bring the burden of public key storage. In [15], the operation of saving the relevant secret value before executing the protocol is omitted. It saves the sender's storage burden. The disadvantage is that it can only resist passive attacks that are not strong enough. The identity-based authenticated asymmetric group key agreement (IB-AAGKA) proposed in the scheme [16] proposes that if the user wants to resist active attacks such as man-in-the-middle attacks, the authentication process should be added to confirm that the identity of the participant is authenticated.

Similar to ASGKA, based on the K-BDHE assumption, one-round of IB-AAGKA protocol is constructed using bilinear mapping. IB-AAGKA is also suitable for the construction

of broadcast encryption. In the **Setup** phase, cloud users run the IB-AAGKA protocol. The confidential security requirements in this protocol ensure that messages encrypted by the negotiated key in the group can only be accessed by legitimate members. For a plaintext database (i, m_i), users can generate ciphertext by computing $c_1 = g^{\rho}, c_2 = w^{\rho}, c_3 = m_i \oplus H^{\rho}(\hat{e}(\prod_{i=1}^{n} H(ID_i), g_1)$, for $\rho \in Z_p^*$. Obtain ciphertext data $c_i = (c_1, c_2, c_3)$. Then the encrypted ciphertext database (i, c_i) is obtained. But to decrypt the information, only the user U_i who has the decryption key that uses the IB-AAGKA protocol to generate the corresponding identity can decrypt the ciphertext c_i.

C. ERROR DETECTION

From the construction of the scheme, we can see that the data in the vector commitment is set as an orderly sequence of data. The data set consists of many data blocks, each of which has its own serial number to represent the location information of the data block. When TPA wants to audit data, TPA can randomly select data from different locations in the data block to audit. This method similar to sampling detection can improve the accuracy and efficiency of detection of data storage. Since the number of checks per time is certainly less than the amount of data detected, the computational overhead in the audit process is also reduced. TPA randomly selects i data blocks in the database with n data blocks for correctness verification. Assuming that only j data blocks have storage errors in n data blocks, and if the number of j is constant then the probability of detecting a server storage error is constant. Therefore, the random sampling detection method derived from the vector commitment *position binding* attribute can effectively detect the storage condition of the server.

V. CORRECTNESS AND SECURITY ANALYSIS

- **Theorem 1.** *Semantic security of linear encryption based on the DLIN assumption our scheme is anonymous.*

  Proof: Suppose $\mathcal{A}$ is an adversary trying to break the anonymity in our scheme, and there exists an algorithm $\mathcal{V}$ that can break the semantic security of the linear encryption scheme obtained from the DILN assumption in the signature scheme. In order to facilitate the narrative, we defined the oracles involved in the simulation security in advance: O_Q can provide the adversary with public information in the system; O_{Join} allows the adversary to get some status information about the user's personal identity through GM and user interaction; O_{sign} outputs the valid signature of the user on a message calculated by the adversary based on the information obtained.

  a. O_Q: Send this public key to algorithm $\mathcal{V}$ as input. Set $P = (p, G, G_T, g_1, g_2, e)$. $\mathcal{V}$ randomly chooses $g, g_1, g_2 \in G, k_m \leftarrow Z_p$, and calculates $k_v = g^{k_m}$. From this, $gpk = (P, g, H_G, H, k_v)$ can be obtained.

  b. O_{Join}: With O_Q, $\mathcal{V}$ returns a counter i about the number of users, and $Q = (p, G, G_T, g_1, g_2, e, u, v, z)$. $\mathcal{V}$ chooses $v_i'', z_i \in_R Z_p$, and calculates S_i' as described above. At this point, a user membership tag is denoted by ind. A more detailed description is $ind_i = (S_i', v_i'', z_i)$, and the membership secret value is $secret_i = (u_i, v_i')$, where $u_i, v_i' \in Z_p$. Then $\mathcal{V}$ adds U_i to the set of users that are not corrupted.Finally, $R_{state} = (U_i, ind_i, secret_i)$ is given to $\mathcal{A}$.

  c. O_{sign}: $\{U_i, ind_i, secret_i\}$ information was obtained by $\mathcal{A}$. Query the entry $\{i, ind_i, sec_i\}$, and use $\{ind_i, sec_i\}$ to generate the signature σ. If such an entry does not exist or the user has lost membership, output $\perp$. That is, the adversary $\mathcal{A}$ cannot track the identity of the user.

  The probability that we set the adversary $\mathcal{A}$ to win is expressed as $ad_{\mathcal{A}}^{DLIN}(\lambda) = |Pr[\mathcal{A}(u, v, w, u^a, u^b, u^{a+b}) = ture] - Pr[\mathcal{A}(u, v, w, u^a, u^b, \xi) = ture]|$. If this probability is negligible. Obviously, based on the DLIN assumption, it is not feasible to solve the difficult problem on G. The anonymity of our signature scheme can be guaranteed.

- **Theorem 2:** *Under the assumption that the decision linear logarithm in G is not feasible, our scheme is non-frameability.*

  Proof: Non-Frameability means that even if an adversary observes the process of a user joining and signing messages as GM, he cannot use this information to calculate user signatures. In this game, we are required to treat the attacker as GM. In this case, our scheme will be threatened by two kinds of threats. The attacker tries to analyze the signature composition in the group and track the source or plunder other's signatures. To clarify that the scheme can resist such attacks, we introduce the following oracles in advance: O_Q and O_{sign} as described above; O_U treats the adversary as GM in this Oracle and can join members through O_U. However, in this case, some information about the user is leaked to the adversary, so this kind of user is also called the corrupted user. Let $\mathcal{V}$ construct an algorithm that can break the q-SDH assumption by using the information of the adversary $\mathcal{A}$.

  a. O_Q: Select $g, g_1, g_2 \overset{R}{\leftarrow} G$, and g_1^{ι}, g_2^{ι} as input to algorithm V. ι is an unknown value. $\mathcal{V}$ tries to find ι to prove that it is an algorithm that can solve the discrete logarithm problem based on G. At this time, the public parameters are $P = (p, G, G_T, g, g_1, g_2, e, H_G, H)$. $\mathcal{V}$ selects $k_m \overset{R}{\leftarrow} Z_p$ as the secret key of the GM. And return the current group public key as $gpk = (p, G, G_T, g, g_1, g_2, e, H_G, H, g^{k_m})$.

  b. O_U: $\mathcal{V}$ selects $u_{i^*}, v_{i^*}' \in Z_p$, and sets $g_1^{u_{i^*}} g_2^{v_{i^*}'} + g_1^{\iota} g_2^{\iota}$ is equivalent to $g_1^{u_i} g_2^{v_i'}$. $\mathcal{V}$ obtains $(S_{i^*}', v_{i^*}'', z_{i^*})$ from $\mathcal{A}$, where $v_{i^*}'', z_{i^*} \overset{R}{\leftarrow} Z_p$. Then there is currently $ind_i \leftarrow (S_{i^*}', v_{i^*}'', z_{i^*})$,$secret_i \leftarrow (u_{i^*} v_{i^*}' + \iota = u_i v_i')$.

Then add $(i, ind_i, secret_i)$ to the current state R_{state}. The user U_i joined by this Oracle is called the corrupted user.

c. O_{sign}: When adversary $\mathcal{A}$ successfully outputs (m, σ^*, Λ^*) through an interactive protocol. Set the user's secret value is $(u_i, v_i) = \alpha_i$. In order to extract the SDH representation $(\hat{S}, \hat{\alpha}, \hat{z})$, it can be realized by the following situation, assuming that any signature generated by the user registered by O_U can be traced, then We can extract the SDH representation. From the $\hat{\alpha} = secret_i = u_{i^*} v'_{i^*} + \iota$, $\mathcal{V}$ can find the unknown secret value $Iota$. Since the verified signature is output by the corrupted user, $\hat{\alpha}$ can be extracted, then $\mathcal{V}$ can find ι.

The probability that we set the adversary $\mathcal{A}$ to win is expressed as $ad_{\mathcal{A}}^{q-SDH}(\lambda) = Pr[\mathcal{A}(g_1, g_2, g_2^u, \ldots, g_2^{u^q}) = (g_1^{1/u+z_i}, z_i) \in Z_p] \geqslant \epsilon$. If the probability ϵ is negligible, our scheme is safe against frame attacks.

- **Theorem 3.** *The cloud will reject all operations on the data from the revoked user.*

  Proof: When the user is corrupted, the group manager can revoke the user and the user's privacy will not be revealed. The revocability of our scheme is derived from the group signature scheme that we adopted. Assume that all users have been provided with a revocation list. When a user wants to upload a file modified or updated by himself. The cloud-first verifies whether he is a group user, and then verifies whether he is a valid group user, that is, he is not revoked. First, verify whether it is a group user by verifying τ, that is to say, verify the validity of the equation $e(C, k_v)/e(g, g) = e(g_1, g)^{u_i} e(g_2, g)^{\varepsilon} e(g_2, k_v)^{\alpha} / e(C, g)^{z_i}$. When it is confirmed that the user is a group user, then it is checked whether it has a valid qualification. Each user generates its own tag invalidation certificate and uses its own secret value u_i, and this secret value is also an important identifier for identifying the tag. If we want to verify whether an invalid tag is generated by this user, and based on the anonymity of our scheme, we can ensure that no one can recognize the real identity of this user except GM with k_m. As described in the above scheme $tag_{i^*} = (\mu = g^{u_{i^*}+\beta}, \zeta = \varphi^{\beta}, \varphi)$. And $\Gamma = g^{\theta(u_i+\delta)}, \Phi = \varphi^{\delta}, \Omega = g^{\theta}, \Upsilon = \varphi^{\theta}$. If the user U_i is revoked, then in the case of $i^* = i$, equation $e(\varphi_j, \Gamma_j)/e(\Phi_j, \Omega_j) = e(\Upsilon_j, \mu_j)/e(\zeta_j, \Omega_j)$ holds. Explain that this tag is generated by U_i, and $tag_{i^*} \in RL$, then the user is revoked. The cloud server will reject any actions the user has made on the data.

- **Theorem 4.** *Only legitimate group users modify the data.*

  Proof: Only valid group users can update the data, etc. Users who want to update data shared by data owners in the cloud should first initiate a registration request with the group manager for secret key $k_u[i]$. The data user U_i selectsu_i, v'_i as the private key from Z_p. Then computes the corresponding public key S_i to send to the group manager GM. GM selects v''_i, z_i from Z_p and computes S''_i and sends (S'_i, v''_i, z_i) to user U_i. U_i computes $v_i = v'_i + v''_i modp$. Whether the user is registered successfully is proved by the following equation:

$$\begin{aligned} e(S'_i, k_v g^{z_i}) &= e\{(S_i g_2^{v''_i} g)^{1/(k_m+z_i)}, g^{k_m+z_i}\} \\ &= e(g_1^{u_i} g_2^{v'_i} g_2^{v''_i} g, g) \\ &= e(g_1^{u_i} g_2^{v'_i+v''_i} g, g) \\ &= e(g_1^{u_i} g_2^{v_i} g, g) \end{aligned}$$

  After verification, the user confirms the parameter from the GM to obtain the secret key. Only a user holding a secret key can generate a valid signature, and a legitimate user in the group can update the message which is stored on the cloud.

- **Theorem 5.** *It is computationally infeasible to generate (m, σ) that can be verified without knowing the secret key.*

  Proof: Assuming that a group of users without collusion is regarded as a collection $\mathcal{U}$, the members of this collection can include all group members. These members can generate signatures that cannot be determined to a specific identity and cannot be resolved. The following games are used to illustrate the unforgeability of signatures in our scheme. The game has two phases: first, $\mathcal{A}$ specifies a group of members, and determines the identity of the group members and their total amount. We use $\mathcal{U}'$ to represent this member collection. Next is the adversary $\mathcal{A}$ try to forge (m_i, σ_i).

  a. O_k: Give GM's secret key k_m and group public key gpk to a PPT adversary $\mathcal{A}$.

  b. O_{sign}: Suppose the adversary selects n messages from the message sequence and queries the signature corresponding to the selected message, wherein the number of times the signature is queried is limited to q_n.Then the challenger sends the signature $\sigma_i = sign(gpk, RL, k_u[i], m_i)$ corresponding to the message he queried to the adversary. Where $sign$ represents the signature algorithm.

  c. O_o: The adversary outputs the message signature pair (m_i, σ_i). If (m_i, σ_i) does not appear between the previously signed message signature pairs. And the message signature pair (m_i, σ_i) can pass the verification then this verification algorithm is expressed as $\text{Verify}(gpk, m_i, \sigma_i) = 1$. It is said that the adversary won the game.

  The probability that we set the adversary to win is expressed as $ad_{sign,\mathcal{A}}(\lambda, n) = Pr[Gam_{sign,\mathcal{A}}(\lambda, n) = 1]$. The probability of $\mathcal{A}$ in winning this game is negligible and the signature used in the scheme is unforgeable.

- **Theorem 6.** *Our audit scheme achieves the correctness of storage.*

  Proof: Given a message set $\mathcal{M}$, the verifier can verify that the data is not corrupted. Knowing a set of

TABLE 1. Performance evalution and comparison.

Scheme	Jiang [5]	Yuan [17]	Wang [18]	Our Scheme
Proof Generation	*(n-1)(M+E)*	*n(M+P+3E)*	*n(E+2M)*	*(n-1)(M+E)*
Verify	*7P+M+9E+5H*	*5E+5P+2H*	*3E+2M+4P+2H*	*E+6P*
ProofUpdate	*2d(M+E)* or *d(M+E)*	*dE+(d+1)M+cP*	*d(E+2M)*	*2d(M+E)* or *d(M+E)*
UserRevocation	*r(M+2P)*	*r(P+E+M)*	*2r(P+H+M+2E)*	*r(H+2E)*

message sequences derived from the message space, we want to generate a corresponding vector commitment on this sequence of messages. And generate proof that this commitment is generated by the current sequence $m_1, \ldots, m_q$ that we have selected. We ensure the correctness and integrity of the data by the following equation:

$$\begin{aligned} e(C_v^t / h_i^{m_i^t}, h_i) &= e(\prod_{j=1}^{n} h_j^{m_j^t} / h_i^{m_i^t}, h_i) \\ &= \prod_{i \neq j}^{n} e(h_j^{m_j^t}, h_i) \cdot e(h_i^{-m_i^t}, h_i) \\ &= \prod_{i \neq j}^{n} e(h_{i,j}^{(m_j - m_i)^t}, h_i) \\ &= \prod_{i \neq j}^{n} e(h_{i,j}^{m_j^t}, g^{z_i}) \\ &= e(\Lambda_i^t, g) \end{aligned}$$

It is obvious that the equation holds only when $\prod_{j=1}^{n} h_j^{m_j} / h_i^{m_i} = \prod_{i \neq j, 1 \leq j \leq n} h_{i,j}^{m_j}$. In this paper, the input of the message is changed to the input of the commitment. More precisely, the verification algorithm verifies that the commitment is correct. In addition, by comparing random locations, we can also see whether the message is complete and not tampered with or lost.

VI. EVALUATION

A. STORAGE OVERHEAD AND FUNCTIONALITY COMPARISON

We analyze their storage overhead from three entities included in the model of this paper. The model of our scheme consists of a user group with GM, a cloud server responsible for data storage, and a trusted third-party TPA. For group users, they only need to save the secret key k_u obtained after registration with the group manager. TPA does not need to store data uploaded by users. Make sure that only legitimate users can modify and update the data, cloud service providers need to save user RL. The size of the revocation list is linear with the number of revoked users, and the number is always smaller than that of group users.

Table 2 compares our scheme with other integrity audit schemes [5], [17], [18] with user revocation in terms of sample auditing, non-framing, anonymity, and resistance to collusion attacks, and shows our advantages. In our scheme, users can anonymously sign data blocks. This property guarantees that no one can identify the real identity of the signer except GM. Therefore, compared to other data auditing schemes, our scheme not only implements effective user revocation but also has the attributes of user identity privacy protection.

TABLE 2. Functionality comparison.

Scheme	[5]	[17]	[18]	Our Scheme
Sampling auditing	Yes	Yes	Yes	Yes
Collusion-resistant	Yes	Yes	No	Yes
Anonymity	No	No	No	Yes
Non-frameability	No	No	No	Yes

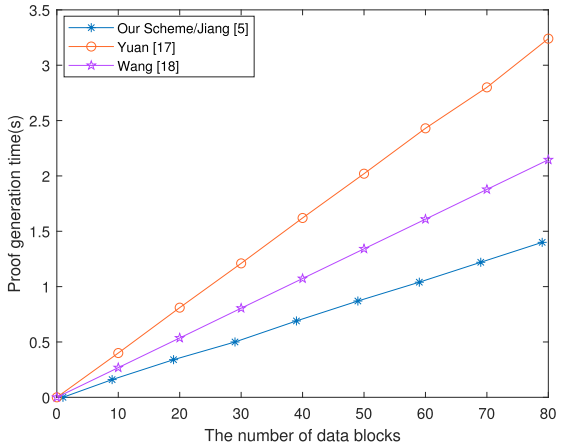

FIGURE 2. Proof generation time.

B. COMPUTING COST

Comparing with schemes [5], [17], [18], to analyze the computational overhead in our scheme. As shown in Table 1. The calculation cost of each scheme can be clearly compared. *M* stands for multiplication on the multiplication cycle group, *P* for pairing, *H* for hash, and *E* for exponentiation. On the client side, after initialization, the group manager executes the key generation algorithm *KeyGen* to generate the group key. After the group user is registered, the signature and the corresponding *tag* are generated. And these computational costs are one-time. Computing about databases is outsourced to cloud servers, so it can save the computing cost of clients. Cloud servers store data uploaded by users, and users do not store data locally to save storage resources.

In our scheme, the proof of the message sequence is obtained by running the **Proof Generation** algorithm. As shown in Figure 2. In this paper, the number of data blocks

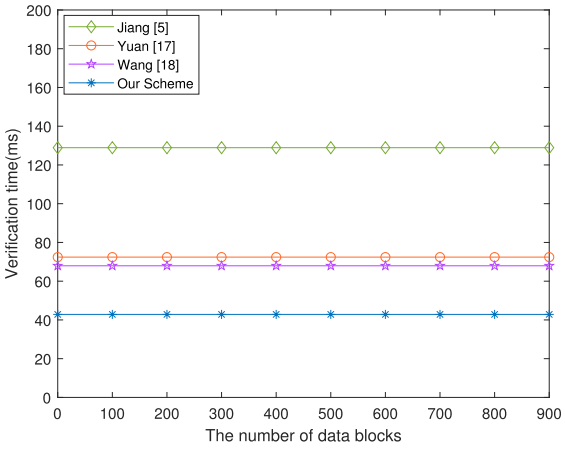

FIGURE 3. Verify time.

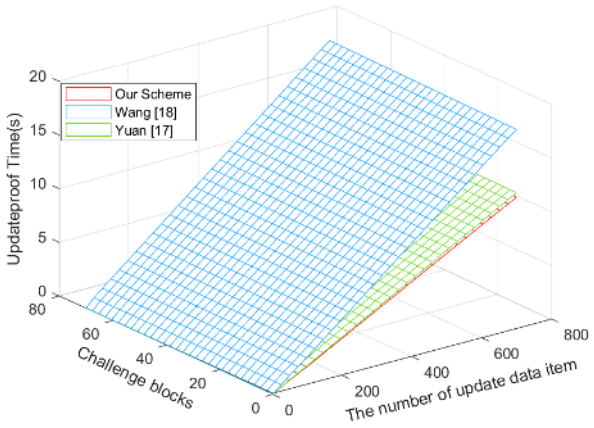

FIGURE 4. ProofUpdate time.

in the message sequence is expressed as n. It is easy to see that the computational time cost of the algorithm will increase with the number of data blocks. However, the server does not need to generate proofs for the entire sequence every time, because it only needs to generate proofs according to the sequence number at the time of query. Previously generated proofs can be cached for later queries when needed. Comparing with Jiang *et al*'s scheme, our scheme saves computational overhead by using short group signature in the computation of the **Verify** algorithm. In the verification phase, we compared the computational overhead of the two parts of data integrity verification and user signature or tag verification. As shown in Figure 3, our verification efficiency is the most efficient compared with schemes [18] and [17], to resist collusion attacks, it is essential in our scheme, and scheme [18] can not resist such attacks. As shown in Figure 5. The computational cost of the **User Anonymous Revocation** algorithm in this scheme is related to the number of challenge blocks c. In the Yuan *et al*. scheme, the computational time cost of user revocation will not increase with the expansion of group members, but the computational time of scheme [18] will increase with the increase of members. In our scheme, the computational time is only related to the number of revoked users. In this phase, it is obvious that the number of revoked users r in the same size group is always less than the number of members d in the whole group. In the time cost analysis, we assume that the number of r is half of the number of d. Wang *et al*'s scheme uses proxy signature technology. The drawback of the scheme is that it can't resist the collusion attack of revoked users with a group signature private key and cloud with the re-signature key. In the user revocation phase, cloud servers are required to help users sign proxy signatures, so our scheme is also effective compared with this scheme.

In the proof updating stage, the main computational overhead of our scheme comes from the updating of vector commitment, that is, the running of the **ProofUpdate** algorithm. Because the sequence of data blocks has changed, the commitment values associated with data blocks need to be recalculated. Therefore, it is the same as the scheme [17]. The computational time spent at this stage is related to the amount of updated data. And we only need to calculate it once when the data changes dynamically. When accessing data, we do not need to update the corresponding data proof. As shown in Figure 4, our proof update efficiency is clearly the best.

C. PERFORMANCE EVALUATION

In our experiments, we utilize Java Pairing Based Cryptography (JPBC) library [19] to simulate, and the experiments are tested on a machine with Intel(R) Core(TM) i5-7300HQ CPU@2.5GHz processor and 8G memory. The computing cost involved in the generation phase of the proof is borne by the cloud server. Compared with the laptop computer used in the experiment, the computing power of the cloud is very powerful. As shown in Figure 2, The **Proof Generation** algorithm only needs to be calculated when the data changes. If only data is accessed, there is no need to recalculate the proof of the data. The computational overhead involved in the verification of proof and signature mainly comes from the verification of group signatures. As shown in Figure 3, compared with schemes [5], [17], [18], our scheme reduces the computational overhead, thus improving the efficiency of verification and reducing the computational time cost.

Since both schemes [5] and our scheme adopt vector commitment to managing proof updating, we only compare the time cost of **ProofUpdate** algorithm with scheme [17], [18] in the updating phase. As shown in Figure 4.Compared with the scheme of [17], [18], our scheme takes less time and computation costs to update proof where the number of data blocks is the same. Although time increases with the amount of updated data, the running time of the algorithm in our scheme is also less than that in scheme [17], [18] when it has the same challenge data block as scheme [17], [18]. Therefore, our scheme is efficient for updating the proof.

As shown in Figure 5, in the user revocation phase, our scheme also has the least computational overhead. In scheme [18], revocation involves the calculation of the whole user group. But the calculation of our scheme and

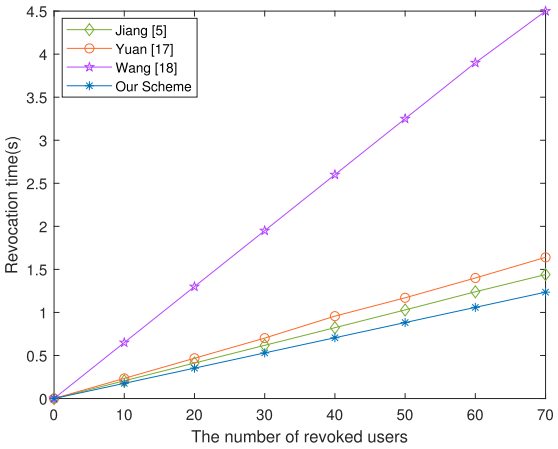

FIGURE 5. Revoked time.

scheme [5], [17] when revoked is only related to the number of revoked users. Scheme [18] needs to be signed by the cloud server proxy during user revocation, so the cost of computing time is higher than other schemes. Our user revocation mechanism is derived from our short anonymous signature the scheme [14]. Compared with scheme [5], the privacy of the user signature in our scheme is also guaranteed.

VII. RELATED WORK

Due to the increasing size of data, it has been previously proven that data-owned scheme requires that no matter how much data is stored, the server needs to traverse the entire data. So it is not suitable for large data volumes. To save the cost of I/O, instead of retrieving the whole data, blocks in the data are extracted to generate proofs to represent the storage of data by the server. This method is called provable data possession (PDP), which was proposed by Ateniese *et al.* [20]. It is the first remote data provable security scheme. The PDP model can verify the integrity of the data but only supports static models. Therefore, it is not practical for a dynamic cloud storage environment. As an improvement scheme [1], the verification of the file identity using the RSA tree authentication dictionary results in an increase in the detection probability. Reference [21] is the first proof based on scheme [22] and also a static PDP scheme. The retrieval proof (POR) proposed by Juels and Kaliski [22]. The prover can prove the integrity of the data stored on his server while verifying that he owns the data to the verifier. It is done by a retrieval method and verifies the correctness of the tags inserted in the data. Because the tag location is fixed, it only supports static data and cannot be completed if the data is dynamically updatable. In order to achieve the purpose of improving this defect, Cash *et al.* [23] first proposed a POR scheme under dynamic data storage by coding data into blocks. Yuan and Yu proposed the first POR-based publicly verifiable and constant communication cost auditing scheme [24]. Reference [25] combines the authentication data structure with the signature scheme to propose a dynamic POR scheme. With the increasing demand for cloud computing, the practical application of cloud storage is more and more extensive. In this environment, the outsourcing data storage auditing scheme has expanded many methods for data integrity verification from the previous one. The first proposed solution is to not support data modification, that is, the integrity verification scheme is static, and then extends the scheme to support dynamic.

Chen and Curtmola proposed a scheme for error detection of dynamically updated data based on dynamic PDP [26]. To achieve the purpose of data privacy protection, Wang *et al.* [8] proposed an audit scheme with data privacy protection and implemented the audit scheme with privacy protection on the basis of the PDP model with random mask technology. Liu *et al.* [27] defined an integrity audit scheme for shared data. Reference [28] utilizing the advantages of cloud storage can save users the cost of building local storage devices and can access resources through mobile terminal devices or webs, greatly improving the flexibility of using data. To prevent deduplication attacks, a malicious attacker who knows the hash signature of a file stored on the server is prevented from obtaining files from the server, [29] introduces the concept of proof of ownership (POW) based on the Merkle tree and specific coding techniques. Besides outsourced data audit, outsourced computation has been well studied [30]–[32]. In order to achieve public verification of data without introducing an additional audit burden for data owners [3], Wang *et al*. introduced third-party auditors to ease the burden on users and implement batch audits of TPA. The related schemes for TPA threats to data privacy [7], [8] As described above, [8] is an extension of [7] to ensure that the TPA cannot obtain any knowledge about the data during the audit. Zhang *et al*. proposed two blockchain-based fair payment protocols called BPay [33] and BCPay [34] for cloud computing applications including data integrity audit. BPay is compatible with the Bitcoin blockchain and BCPay is compatible with the Ethereum blockchain.

Scheme [35] combines fragment techniques and index hash tables with random sampling to ensure data integrity and support data updates and anomaly detection. For the access rights of data, the permissions of the users in the different stages of the project should also be changed accordingly, and some users will inevitably be revoked. Therefore, it also caused the qualification to revoke the user rights management problem. Jiang *et al.* [5] proposed ciphertext storage and prevention of collusion between cloud servers and revoked members. Reference [36] solved the problem that the client's verification of data is not affected by the input size of data. In scheme [37], Yan *et al*. proposed a Hash-based Remote Data User Check Protocol (RDPC) to ensure that data would not be abused. Yan *et al*. in scheme [38] also proposed RDPC scheme with designated verifier function to designate specific user to check data. Li *et al*. also proposed a data integrity verification scheme [39] using certificateless signature technology based on the RDPC protocol. Scheme [40], which can support data state tracking and add user privacy protection function, is also proposed.

As for group signature, each group member generates a signature relative to the group public key according to the secret key of the group. The group administrator cannot create the identity of the group member while holding the secret key without providing the entity. Anonymity is provided for the signature, but the identity of the group member, that is, the traceability of the signature, can be traced while the signature is being given. Reference [5] provides a security definition for these cryptographic primitives. Authentication with revocation has also been used to address security and privacy issues in 5G [41]. The anonymous revocation group signature scheme proposed by Nakanishi and Funabiki [14] is a short group signature scheme that can implement the revocation of users without the privacy manager. Compared with the scheme proposed by Brickell and Li [42], the signature length can be greatly reduced. The signature scheme based on bilinear pairing proposed in scheme [43] greatly reduces the length of the signature. The paper [44] also gives some foundations for constructing signature schemes, such as the definition of formal signature.

VIII. CONCLUSION

In this paper, an efficient sharing auditing scheme for cloud storage data that supports anonymous revocation of users is proposed. By combining vector commitment primitives with an efficient signature scheme and key agreement protocol, we implement user security revocation and prevent collusion between the revoked user and the malicious cloud server. Our scheme supports dynamic changes in data and the joining or exiting of group members. Stored data is not leaked to TPA or invalid users during auditing or sharing. Through the comparison of experimental results, our solution is effective and reduces the computational overhead of audit phases.

REFERENCES

[1] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, Nov. 2009, pp. 213–222.
[2] P. S. Kumar, R. Subramanian, and D. T. Selvam, "Ensuring data storage security in cloud computing using Sobol sequence," in *Proc. 1st Int. Conf. Parallel, Distrib. Grid Comput. (PDGC)*, Oct. 2010, pp. 217–222.
[3] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, May 2011.
[4] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th Int. Conf. Secur. Privacy Commun. Netw.*, Sep. 2008, Art. no. 9.
[5] T. Jiang, X. Chen, and J. Ma, "Public integrity auditing for shared dynamic cloud data with group user revocation," *IEEE Trans. Comput.*, vol. 65, no. 8, pp. 2363–2373, Aug. 2016.
[6] J. Yuan and S. Yu, "Public integrity auditing for dynamic data sharing with multiuser modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 8, pp. 1717–1726, Aug. 2015.
[7] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proc. IEEE Infocom*, Mar. 2010, pp. 1–9.
[8] 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.
[9] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2130–2145, Jun. 2018.
[10] J. Sun, S. Hu, X. Nie, and J. Walker, "Efficient ranked multi-keyword retrieval with privacy protection for multiple data owners in cloud computing," *IEEE Syst. J.*, to be published. doi: 10.1109/JSYST.2019.2933346.
[11] X. Chen, F. Zhang, W. Susilo, H. Tian, J. Li, and K. Kim, "Identity-based chameleon hashing and signatures without key exposure," *Inf. Sci.*, vol. 265, no. 5, pp. 198–210, 2014.
[12] Y. Zhang, R. Deng, D. Zheng, J. Li, P. Wu, and J. Cao, "Efficient and robust certificateless signature for data crowdsensing in cloud-assisted industrial IoT," *IEEE Trans. Ind. Informat.*, to be published.
[13] D. Catalano and D. Fiore, "Vector commitments and their applications," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2013, pp. 55–72.
[14] T. Nakanishi and N. Funabiki, "A short anonymously revocable group signature scheme from decision linear assumption," in *Proc. Acm Symp. Inf., Comput. Commun. Secur.*, Mar. 2008, pp. 337–340.
[15] Q. Wu, Y. Mu, W. Susilo, B. Qin, and J. Domingo-Ferrer, "Asymmetric group key agreement," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.* Berlin, Germany: Springer, 2009, pp. 153–170.
[16] L. Zhang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based authenticated asymmetric group key agreement protocol," in *Proc. Int. Comput. Combinatorics Conf.* Berlin, Germany: Springer, 2010, pp. 510–519.
[17] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," in *Proc. IEEE Conf. Comput. Commun.*, Apr./May 2014, pp. 2121–2129.
[18] B. Wang, B. Li, and H. Li, "Panda: Public auditing for shared data with efficient user revocation in the cloud," *IEEE Trans. Serv. Comput.*, vol. 8, no. 1, pp. 92–106, Jan./Feb. 2015.
[19] A. De Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun./Jul. 2011, pp. 850–855.
[20] G. Ateniese, R. C. Burns, R. I. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, Oct. 2007, pp. 598–609.
[21] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2008, pp. 90–107.
[22] A. Juels and B. S. Kaliski, Jr., "Pors: Proofs of retrievability for large files," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, pp. 584–597, Oct. 2007.
[23] D. Cash, A. Küpçü, and D. Wichs, "Dynamic proofs of retrievability via oblivious RAM," *J. Cryptol.*, vol. 30, no. 1, pp. 22–57, Jan. 2017.
[24] J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud," in *Proc. Int. Workshop Secur. Cloud Comput.*, May 2013, pp. 19–26.
[25] Q. Zheng and S. Xu, "Fair and dynamic proofs of retrievability," in *Proc. 1st ACM Conf. Data Appl. Secur. Privacy*, Feb. 2011, pp. 237–248.
[26] B. Chen and R. Curtmola, "Robust dynamic provable data possession," in *Proc. 32nd Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2012, pp. 515–525.
[27] C.-W. Liu, W.-F. Hsien, C.-C. Yang, and M.-S. Hwang, "A survey of public auditing for shared data storage with user revocation in cloud computing," *Int. J. Netw. Secur.*, vol. 18, no. 4, pp. 650–666, Jul. 2016.
[28] C. Liu, R. Ranjan, C. Yang, X. Zhang, L. Wang, and J. Chen, "MuR-DPA: Top-down levelled multi-replica Merkle hash tree based secure public auditing for dynamic big data storage on cloud," *IEEE Trans. Comput.*, vol. 64, no. 9, pp. 2609–2622, Sep. 2015.
[29] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in *Proc. 18th ACM Conf. Comput. Commun. Secur.*, Oct. 2011, pp. 491–500.
[30] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, "Verifiable computation over large database with incremental updates," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 3184–3195, Oct. 2016.
[31] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, "New publicly verifiable databases with efficient updates," *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 5, pp. 546–556, Sep./Oct. 2015.
[32] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2386–2396, Sep. 2014.
[33] Y. Zhang, R. Deng, X. Liu, and D. Zheng, "Outsourcing service fair payment based on blockchain and its applications in cloud computing," *IEEE Trans. Services Comput.*, to be published.
[34] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Blockchain based efficient and robust fair payment for outsourcing services in cloud computing," *Inf. Sci.*, vol. 462, pp. 262–277, Jun. 2018.

[35] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE Trans. Services Comput.*, vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.
[36] M. Backes, D. Fiore, and R. M. Reischuk, "Verifiable delegation of computation on outsourced data," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2013, pp. 863–874.
[37] H. Yan, J. Li, J. Han, and Y. Zhang, "A novel efficient remote data possession checking protocol in cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1, pp. 78–88, Jan. 2017.
[38] H. Yan, J. Li, and Y. Zhang, "Remote data checking with a designated verifier in cloud storage," *IEEE Syst. J.*, to be published.
[39] J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Trans. Services Comput.*, to be published.
[40] 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*, to be published.
[41] Y. Zhang, R. Deng, E. Bertino, and D. Zheng, "Robust and universal seamless handover authentication in 5G HetNets," *IEEE Trans. Dependable Secure Comput.*, to be published.
[42] E. Brickell and J. Li, "Enhanced privacy ID: A direct anonymous attestation scheme with enhanced revocation capabilities," in *Proc. ACM Workshop Privacy Electron. Soc.*, Oct. 2007, pp. 21–30.
[43] S. G. Choi, K. Park, and M. Yung, "Short traceable signatures based on bilinear pairings," in *Proc. Int. Workshop Secur.* Berlin, Germany: Springer, 2006, pp. 88–103.
[44] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2003, pp. 614–629.

YINGHUI ZHANG (M'18) has been a Professor with the National Engineering Laboratory for Wireless Security (NELWS), Xi'an University of Posts and Telecommunications, since 2018. He has published over 80 research articles in ACM ASIACCS, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, *Computer Networks*, the IEEE INTERNET OF THINGS JOURNAL, *Computers & Security*, and the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS. His research interests include public key cryptography, cloud security, and wireless network security.

CHEN CHEN received the B.S. degree from the Xi'an University of Posts and Telecommunications, Xi'an, China, in 2016, where she is currently pursuing the M.S. degree. Her research interest includes cloud storage security.

DONG ZHENG received the Ph.D. degree in communication engineering from Xidian University, China, in 1999. He was a Professor with the School of Information Security Engineering, Shanghai Jiao Tong University. He is currently a Professor with the National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications. He has published over 100 research articles, including CT-RSA, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, and *Information Sciences*. His research interests include cloud computing security and public key cryptography.

RUI GUO received the Ph.D. degree from the State Key Laboratory of Networking and Switch Technology, Beijing University of Posts and Telecommunications, China, in 2014. He is currently a Lecturer with the National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications. His current research interests include attribute-based cryptograph, cloud computing, and blockchain technology.

SHENGMIN XU received the B.Sc. degree from the School of Computing and Information Technology, University of Wollongong, Australia, in 2014, and the Ph.D. degree in cryptography from the University of Wollongong, Australia, in 2018. He is currently a Research Fellow with the School of Information System, Singapore Management University, Singapore. His research interests include cryptography and information security.