IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ADVANCED COMMUNICATIONS AND NETWORKING TECHNIQUES FOR WIRELESS CONNECTED INTELLIGENT ROBOT SWARMS

Received March 27, 2020, accepted April 25, 2020, date of publication April 28, 2020, date of current version May 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2991033

Privacy-Preserving Public Cloud Audit Scheme Supporting Dynamic Data for Unmanned Aerial Vehicles

JIASEN LIU[1], XU AN WANG[1], ZHIQUAN LIU[2,3], HAN WANG[1], AND XIAOYUAN YANG[1]

[1]Key Laboratory for Network and Information Security of the PAP, Engineering University of the People's Armed Police (PAP), Xi'an 710086, China
[2]College of Cyber Security, Jinan University, Guangzhou 510632, China
[3]Guangdong Key Laboratory of Data Security and Privacy Preserving, Jinan University, Guangzhou 510632, China

Corresponding author: Xu An Wang (wangxazjd98@vip.163.com)

This work was supported in part by the National Cryptography Development Fund of China under Grant MMJJ20170112, in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2018JM6028, in part by the National Natural Science Foundation of China under Grant 61772550, Grant U1636114, Grant 61802146, and Grant 61572521, and in part by the National Key Research and Development Program of China under Grant 2017YFB0802000, in part by the Fundamental Research Funds for the Central Universities under Grant 11618332, in part by the Natural Science Foundation of Guangdong Province under Grant 2018A030313813, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515011017, and in part by the Engineering University of People's Armed Police (PAP) Funding for Scientific Research Innovation Team under Grant KYTD201805.

ABSTRACT As one of the most popular applications in recent years, cloud storage has been gradually integrated into all walks of life. In the field of communication techniques for the unmanned aerial vehicles (UAVs), UAVs use sensors to upload collected data to cloud servers during various exploration activities, but UAVs can only store and calculate valid information currently collected due to the limited storage and computing performance. In the actual exploration, UAVs need not only to upload complete data to the cloud server, but also to support the data dynamic update efficiently. Moreover, due to security requirements, the privacy of data uploaded by UAVs must be protected. However, the existing auditing schemes for dynamic data and integrity have many problems, such as high computation cost, low efficiency of dynamic update, low privacy and security. For this reason, we propose a public cloud audit scheme that supports dynamic data and privacy protection based on distributed string equality check protocol and Merkle-hash tree multi-level index structure. First of all, a third-party server (TPS) is set between the cloud service provider and users, which complete digital signature, integrity auditing, and data dynamic operations significantly in place of users to reduce the local computing cost. Users then locally upload data which has been encrypted to the TPS. Secondly, to further improve the security of the scheme, TPS implement signature for encrypted data based on the distributed string equality check protocol. By designing authorizations with time constraints, it is guaranteed that only the legitimate TPS with time constraints can operate with cloud servers. Finally, we implement dynamic data operation efficiently based on MHT multi-level index structure. The security proof and performance analysis show that our proposed scheme is safe and effective.

INDEX TERMS Cloud Storage, communication techniques for UAVs, privacy-preserving, dynamic updating, intergrity auditing.

I. INTRODUCTION

With the arrival of the era of big data, cloud computing [22], [23] has become a hot spot for people to research and apply. It decomposes huge data calculation processing programs into countless applets, which are analyzed and processed by distributed servers, and ultimately returns the results to users. As one of the core services of cloud computing, the cloud storage is purposed to provide users with secure, reliable, high-performance and low-cost data storage services. With the development of technology and the society, more and more companies and individuals are choosing to outsource data to cloud service providers to reduce local storage costs.

At the same time, with the development of communication technology and wireless network [15]–[17] unmanned aerial vehicle is gradually being applied to all walks of life in

The associate editor coordinating the review of this manuscript and approving it for publication was Venanzio Cichella.

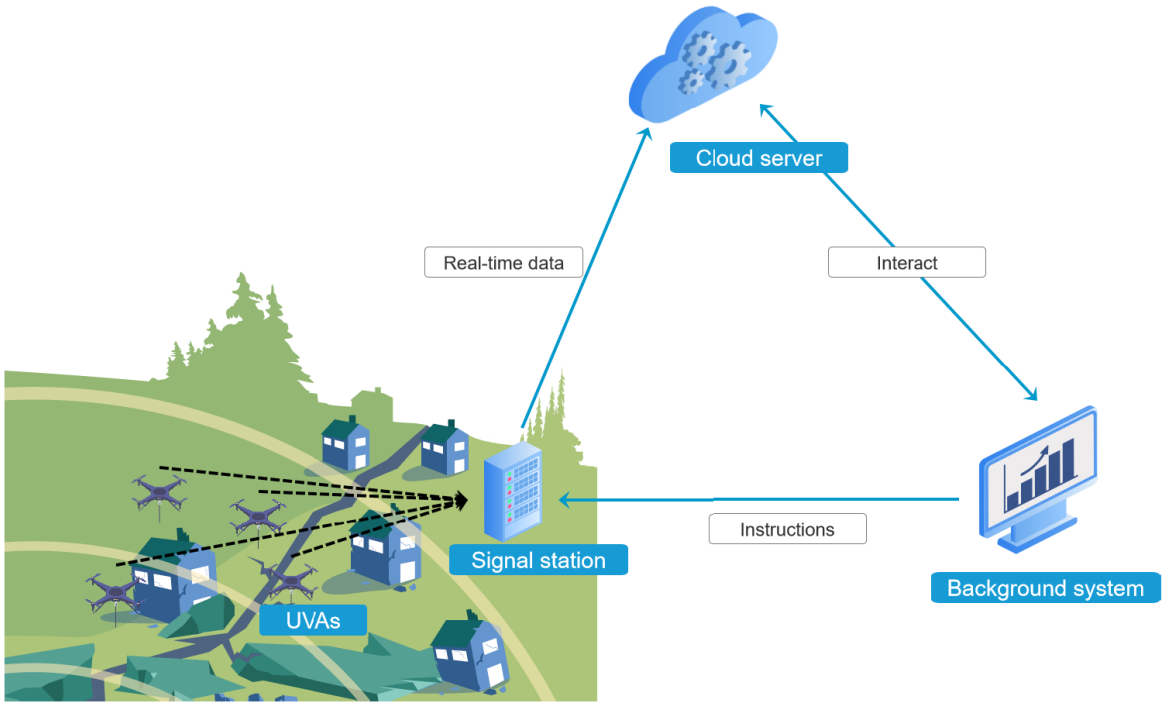


FIGURE 1. Data transmission in exploration.

society. In recent years, scientific and technological news about UAV exploration has often appeared in news reports. UAVs upload real-time data collected by sensors to cloud servers, and provide real-time analysis of data behind the scenes. People can make the initial flight trajectory of an unmanned air fleet and modify it instantly based on the real-time data. Data transfer activities for this process can be shown in Figure 1:

However, it is important to note that this process can ensure the data uploaded by the UAV fleet is complete and can be dynamically updated in real time according to actual conditions. Unfortunately, the computing and storage capabilities of the UAVs are limited by their limited size. They can only support the storage of the real-time data, as well as the simple encryption of current data. Therefore, a public audit scheme supporting dynamic data is needed to accomplish this process. In the current scenario, in order to reduce the user's computing overhead in the audit process, a third-party auditor (TPA) is usually introduced to complete the tedious audit process instead of the user. TPA can verify the integrity of outsourced data through proven data holding (PDP) and retrievable proof (POR).

In order to efficiently and safely realize dynamic data integrity auditing with privacy-preserving in cloud storage system, this paper is set out to realize a dynamic public auditing based on the distributed string equality check protocol and hierarchical index structure. At the same time, in order to better realize the privacy protection of outsourced data and reduce the computing and communication overhead of users, we set up an intermediate node TPS between users and cloud service providers, which is responsible for signing the Encrypted data uploaded by users and completing the integrity audit for the users. In addition, users need to grant TPS an authentication that is only valid within a specified time interval before each operation, so as to assure the security of TPS.

The scheme not only realizes the dynamic operation and meets the security requirements, but also further protects the privacy of data and reduces the computing overhead of users. In summary, the scheme in this paper can achieve the following objectives:

1) Support multi-granularity dynamic operation of outsourcing data.
2) Implement public integrity audit of outsourced data under privacy protection.
3) Reduce the user's computation and communication overhead.
4) Set up a security node TPS to reduce the computing overhead of the UAV through proxy signature. Security and reliability are guaranteed through authorization sent to TPS by unmanned aerial vehicles.

The rest of this article is in full description below. In the second chapter, we introduce the system model and security objectives of our scheme, and in the third chapter, we introduce our scheme in detail. In the fourth chapter, we analyze the effectiveness, security and performance of the system scheme. Finally, we summarize the article in the fifth chapter.

II. RELATED WORK

To evaluate cloud storage system security, the integrity and availability of outsourced data are necessary indicators. In order to ensure the integrity and availability of outsourced data, many studies have been carried out around data integrity auditing [1]–[12], [24]–[36]. As one of the core feature of cloud storage system, integrity auditing allows users to verify whether the uploaded data is complete and available, instead of download the cloud data. Data integrity verification mechanisms can be divided into PDP (Provable Data Possession) and POR (Proofs of Retrievability) based on whether fault-tolerant preprocessing is applied to data files. PDP can quickly verify the integrity of outsourced data, and POR can also recover damaged data. To verify the integrity of data in cloud servers, Ateniese *et al.* [1] first proposed a PDP mechanism to verify the integrity of data on untrusted cloud servers. They complete data integrity verification based on the RSA signature mechanism and probabilistic strategy. Their scheme supports public auditing. Based on RSA signature, cloud servers can aggregate proof information to reduce communication overhead. Since then, many PDP schemes have been proposed. Some schemes are based on basic number theory and some are based on elliptic curves cryptograghy. Chen *et al.* [12]. implemented data integrity auditing based on distributed string equality check protocol. Although the efficiency and security of the scheme are relatively high, but public auditing is not supported.

However, for the above schemes, dynamic operation of data can not be supported. The schemes of dynamic data are required to the insertion, deletion and modification of data. At present, many data integrity audit schemes for cloud storage services have been proposed. Giuseppe Ateniese *et al.* [1] proposed a PDP scheme that supports dynamic data based on symmetric cryptosystem in 2007, but the scheme cannot support data insertion. Since then, basing on different authentication structures, different dynamic data auditing schemes have emerged: some are implemented based on index Hash table [2], [21], while some are implemented based on Merkle Hash tree (MHT) [18]–[20]. However, the ultimate common goal of these schemes is to improve the operation efficiency of dynamic data and effectively resist any replay, forgery and deletion attacks. Erway *et al.* [5] realized dynamic data operations merely through rank-based authentication jump table dynamic data structures and RSA signature mechanisms. The scheme is the first PDP mode that supports the complete dynamic data operation. However, with the increase of the file block size, the time consumption of node search increases sharply while the dynamic operation efficiency decreases. At the same time, Tian *et al.* [2] proposed a new public audit scheme to secure cloud storage based on Dynamic Hash Table (DHT). It realizes dynamic data integrity auditing by establishing a dynamic two-dimensional data structure on a third-party auditor (TPA). For the efficiency of dynamic data operation, Wang *et al.* [3] proposed a dynamic data integrity auditing scheme that supports public auditing based on MHT and BLS signature mechanisms. In this model, MHT structure is used to ensure the spatial accuracy of blocks. In a recent study, Shao *et al.* [4] and Fu *et al.* [26] consulted Wang *et al.* [3] to implement dynamic data auditing by building hierarchical binary trees (HMBT and MPHT), respectively. Gan *et al.* [6] constructed a new data structure, the record table (RTable), to operate on dynamic data. They implement integrity auditing based on algebraic signatures and XOR homology functions. Aujla *et al.* [25] used grid method and Bloom filter method to verify dynamic data integrity and can resist the attack of quantum computers. Aujla *et al.* [25] constructed a dynamic POR scheme using trapdoor commissions.

However, it is important to note that the above schemes complete the integrity audit by set a third party audit (TPA) to, in order to reduce the user's local computing burden. However, this scenario presents a new security issue because it may leak the privacy of user data. As far as data privacy protection is concerned, users need to prevent TPA (even cloud servers) from obtaining real data from certain data with high confidentiality. At present, the proposed schemes usually use BLS signature scheme and homomorphic linear authenticator to ensure the privacy of the data to TPA, but they do not support dynamic data operation. In addition, their security and efficiency are low.

Wang *et al.* [7] implemented a public cloud data audit system with privacy protection based on random homomorphic authenticator in 2010. Subsequently, in 2014, Worku *et al.* [8] pointed out the source of security defects in Wang *et al.* [7], and further analyzed its inefficiency, then improved it. After that, Yang and Xia [9] and Hong *et al.* [10] respectively conducted researches on privacy protection through elliptic curve cryptography (ECC) and homomorphic encryption schemes for SMC problems based on random mask. In recent research, Xu *et al.* [27] proposed a scheme based on blockchain, which combined with homomorphic encryption and smart contract technology based on Ethereum, and solve the privacy protection problem of electronic health files. Then, Yang *et al.* [28] proposed a attestation-based data access identifying scheme for data confidentiality and design a special log called attestation in which hash user pseudonym is used to preserve user privacy.

III. MODELS AND GOALS

A. SYSTEM MODEL

This section describes the structure of the system model and the functions of each entity. As shown in Figure 2, the system model includes three entities: User (cu), Cloud Service Provider (CSP) and Third-party Server (TPS).

CU includes UAVs and background staff, which are required to upload the real-time acquired data, realize dynamic data operate, verify the integrity of the data and authorize the TPS. The cloud service provider provides users with cloud storage services and other operational

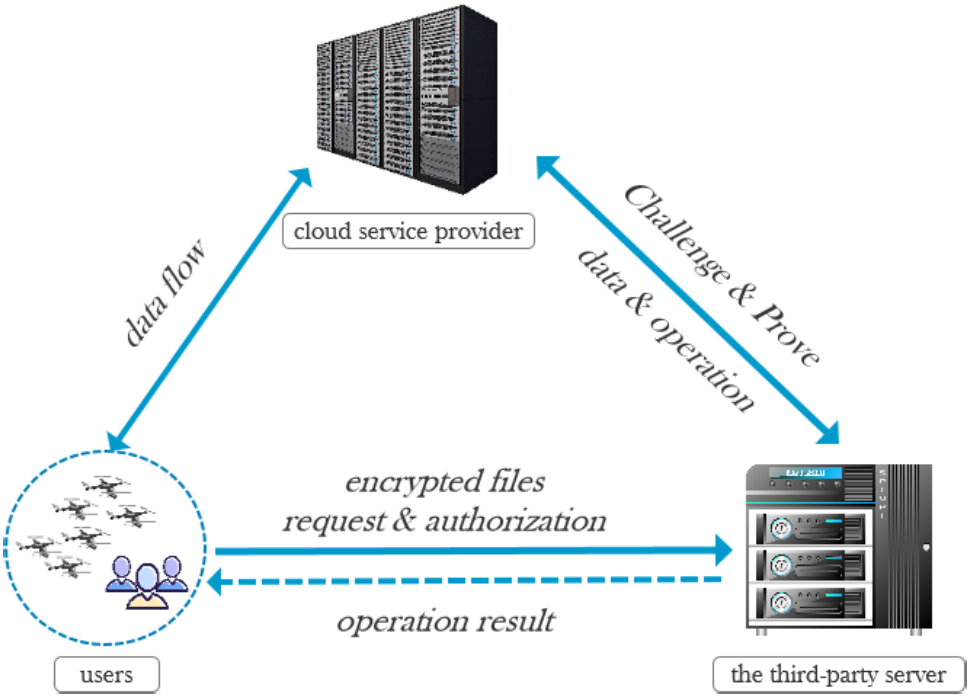


FIGURE 2. Scheme Model.

TABLE 1. Functionality comparison with existing related schemes.

	Public Audit	Dynamic Data	Privacy-Preserving	Data Structure
Anteniese et al. [1]	No	No	No	-
Tian et al. [2]	Yes	Yes	Yes	DHT
Wang et al. [3]	Yes	Yes	No	MHT
Shao et al. [4]	Yes	Yes	Yes	HMBT
Wang et al. [7]	Yes	No	Yes	-
worku et al. [8]	Yes	No	Yes	-
Chen et al. [12]	No	Yes	No	Table
Anteniese et al. [14]	No	Yes	No	Hash function
Fu et al. [26]	Yes	Yes	No	MPHT
Xu et al. [27]	Yes	No	Yes	-
Our scheme	Yes	Yes	Yes	MHT

requirements. TPS is a third-party server. Users send their encrypted data and give authorization to TPS. TPS computes digital signature, sends them to CSP within the valid time limit of authorization, and completes the data integrity audit. Users also send data, operation requests and give authorization to TPS in dynamic operation, then TPS completes the dynamic update and integrity audit of the updated data.

In the system model, CU first generates its own private and public key—key of TPS and authorization. CU encrypts the data with its own private key and sends the encrypted data, public key, key of TPS and authorization to TPS. TPS then signs the data and uploads it to CSP within the authorized effective time. When CU needs to query the integrity of data or to carry out dynamic data operation, CU sends operation application and authorization to TPS. TPS then sends operation request and uploads data to CSP within the effective time of authorization, and audits the integrity of the data through the proof information responded by CSP. Finally, TPS sends the operation result to CU.

B. SECURITY MODEL

Since cloud service providers and TPS are untrusted or semi-trusted, we have listed the following security issues that will occur during integrity auditing and dynamic operations:

1) In order to improve its storage efficiency, CSP maliciously deletes part of the user's data and calculates an aggregated data block and label in advance to pass the integrity audit.
2) TPS or CSP maliciously sells highly confidential data.
3) TPS send a large number of operation requests to the CSP in a short time to consume the communication resources of the CSP.

TABLE 2. Notations.

Symbols	Description
G_1, G_2	multiplicative cyclic group of order
g	generator of
$e: G_1 \times G_1 \to G_2$	bilinear mapping
h, H_1, H_2	Hash functions
f	random function
$m_{i,j}$	data blocks
PK_{CU}	public key of the CU
SK_{CU}	private key of the CU
SK_{TPS}	private key of the TPS
PK_{TPS}	public key of the TPS
Au	authorization
α_i	blinding factor
$name$	unique identification of file
v_i	initial vector of data block
m'_{ij}	encrypted data
σ_{ij}	homomorphic tag
$chal = \{i, e_i\}_{i \in I}$	challenge information
Δ	auxiliary information
$Proof$	proof information
P_R	corresponding information

4) In the dynamic data operation, CSP dishonestly updates the data and deceives the user's data integrity by forging or using expired data and labels.

IV. CONSTRUCTION OF MODEL

A. PRELIMINARIES AND NOTATION

Referring to Chen's scheme [12], this paper combines distributed string equality checking protocol with bilinear pairing to sign the data after privacy protection. Referring to Qing's scheme, multi-granularity dynamic data operation is realized based on MHT. The security of the scheme is realized through Diffie-Hellman problem and pseudo-random function.

1) Bilinear mapping: and G_T are both cyclic multiplicative groups of prime order p, and g is the generator of group G. Then for $\forall m, n \in G, x, y \overset{R}{\leftarrow} Z_p^*$, there is $e(m^x, n^y) = e(m, n)^{xy}$, and $e(g, g) \neq 1$.

2) Diffe-Hellman problem [13]: G is a cyclic group of prime order p, and g is the generator of group G. If (g, g^x, g^y) is given where $x, y \overset{R}{\leftarrow} Z_p^*$, then a cannot be calculated. If (g, g^a) is given where $a \overset{R}{\leftarrow} Z_p^*$, then a cannot be calculated.

3) Merkle Hash Tree: As shown in figure 3, each node structure is $(r_x, h(x))$, where r_x represents the number of data blocks that can be accessed from the node, and $h(x)$ represents the Hash value for verifying the two child nodes. For example, $h(d) = h(h(v_3)||h(v_4))$, and $r_d = r_3 + r_4$. If the auxiliary path is $K = \{J, B\}$, the root node A can be calculated by node K and E, i.e. $h(a) = h(h(b)||h(v_3)||h(v_4))$ and $r_a = r_b + (r_{v_3} + r_{v_4})$.

4) Classic string equality checking protocol [12]: Alice owns strings $x \in \{0, 1\}^n$, Bob owns string $y \in \{0, 1\}^n$, and there is a public random strings pool $S \subseteq \{0, 1\}^n$. Alice selects an $s \overset{R}{\leftarrow} S$ and sends $(s, < x, s > \text{mod} 2)$ to Bob. Bob calculates and verifies $< y, s > \text{mod} 2 \overset{?}{=} < x, s > \text{mod} 2$, and continues if the equation holds, otherwise Bob notifies Alice to terminate the protocol. After repeating this 100, if the above equation holds all the time, Bob notifies Alice that the two strings are equal. The probability of false positives is $1/2^{100}$, which is negligible, and the communication overhead is $O(\log n)$, so the protocol is safe and effective.

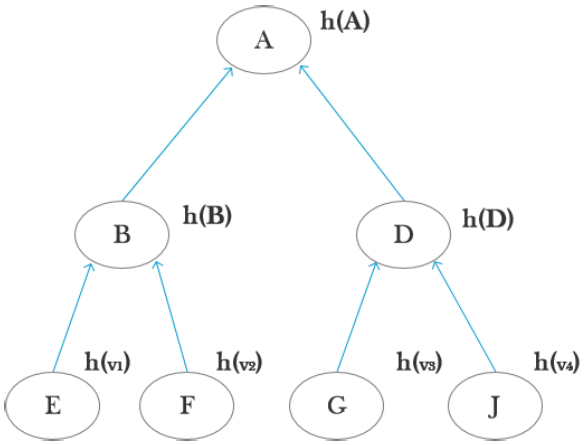


FIGURE 3. Merkle Hash Tree Structure.

B. OUR CONSTRUCTION

In this section, we give a detailed introduction to the system model. G_1, G_2 is both a multiplicative cyclic group of order p, where g is the generator of G_1, and $e: G_1 \times G_1 \to G_2$ is a bilinear mapping. Let $h: G_1 \to Z_p^*, H_1: \{0, 1\}^* \times G_1 \to Z_p^*$, $H_2: \{0, 1\}^* \to G_1$ represent Hash functions. Let $f: Z_p^* \times Z_p^* \to Z_p^*$ be expressed as a random function. The file F is divided into n data partitions $F \to (m_1, \ldots, m_n)$, and each partition m_i is divided into s data blocks $m_i \to (m_{i1}, \ldots, m_{i2})$.

a: SETUP

In this step, CU generates public-private key pairs and authorizations for itself and TPS.

1) The CU selects two random numbers $x, k_1 \overset{R}{\leftarrow} Z_p^*$ as the private key of the CU, and calculates $pk = g^x$ as the public key of the user. Then the CU selects $\varepsilon, r, k_2 \overset{R}{\leftarrow} Z_p^*$ as the private key of the TPS, selects the random number $u \overset{R}{\leftarrow} Z_p^*$ and $k = g^\varepsilon$ as the public key of the TPS. Then the public-private key pair of the user is:

$$PK_{CU} = \{g, pk\}, SK_{CU} = \{x, k_1\} \tag{1}$$

The public-private key pair for TPS is:

$$PK_{TPS} = \{g, k, u\}, SK_{TPS} = \{\varepsilon, r, k_2\} \tag{2}$$

2) CU selects the random number $r_0 \overset{R}{\leftarrow} Z_p^*$, records the valid time interval of the authorization as $[time_1, time_2]$, and then calculates $Y_0 = g^{r_0}$ and $\beta_0 = r_0 + x \cdot H_1(ID_{CU}||ID_{TPS}||time_1||time_2, Y_0) \bmod p$. the CU sends the authorization $Au = \{(ID_{CU}, ID_{TPS}, time_1, time_2), Y_0, \beta_0\}$ and SK_{TPS} to the TPS.

1) DataBlind

CU first carries out privacy protection processing on outsourced data. CU uses private key k_1 to generate

pseudo-random function $f_{k_1}(\cdot)$ and calculates blinding factor $\alpha_i = f_{k_1}(i, name)$, where $name \overset{R}{\leftarrow} Z_p^*$ is the unique identification of file F. After that, the user encrypts the data block and calculates $m'_{ij} = m_{ij} + \alpha_i, i \in [1, n], j \in [1, s]$. The CU sends the encrypted data $F' = \{m'_i\}$ to the TPS.

2) AuthGen

After receiving the file data and authorization. For each data partition m'_i, TPS constructs an initial vector $\mathsf{v}_i = (h(m'_{i1}), \ldots, h(m'_{is}))$ where records the Hash value of each data block m'_{ij} by element v_{ij}, and calculates the homomorphic tag $\sigma_{ij} = (g^{r \cdot m'_{ij} + f_{k_2}(name||i||j||v_{ij})} \cdot u^{H_2(name||i||j||v_{ij})})^{\varepsilon} \bmod p$. After that, TPS constructs MHT and calculates the root node R and the signature $\sigma_R||name$ of R. Finally, the TPS sends $(F', \Gamma, \sigma_R||name)$ to the CSP, where $\Gamma = \{\sigma_{ij}\}, i \in [1, n], j \in [1, s]$ is the set of digital signatures.

When the CSP receives the data, it first verifies the authorization of the TPS by the equation 3:

$$g^{\beta_0} \overset{?}{=} Y_0 \cdot pk^{H_1(ID_{CU}||ID_{TPS}||time_1||time_2, Y_0)} \tag{3}$$

If equation 1 holds, then CSP preserves $(F', \Gamma, \sigma_R||name)$ and constructs corresponding MHT. Otherwise, this operation request is invalid.

3) ProofGen

When conducting integrity audit, CU first authorizes TPS. After the TPS is authorized, it selects a set $I \subseteq [1, n]$ with c elements, and for each i there is $e_i \overset{R}{\leftarrow} Z_p^*$. Than TPS send the challenge information $chal = \{i, e_i\}_{i \in I}$ and the authorize to the CSP.

After the CSP accepts the request, it first verifies whether Authorization Au is valid with $AuthGen$, and then calculates $\alpha = \sum_{i=1}^{I} \sum_{j=1}^{s} m'_{ij} \bmod (p-1)$ and $\beta = \prod_{i=1}^{I} \prod_{j=1}^{s} \sigma_{ij}^{e_i} \bmod p$. If the leaf node auxiliary path of the l-th MHT is K_l, the auxiliary information is $\Delta = \{\mathsf{v}_l, K_l\}_{i \in I}$. Then the CSP sends $Proof = \{\alpha, \beta, \Delta, \sigma_R||name\}$ to the TPS as proof information.

4) ProofVerify

After receiving the certification information, TPS constructs MHT using I and Δ, calculates root node R', and verifies whether $\sigma_R||name$ is valid. If the signature is valid, the TPS calculates $\eta = u^{\sum_{i=1}^{I} \sum_{j=1}^{s} e_i H_2(name||v_{ij})} \bmod p$ and $y = g^{r \cdot \alpha + \sum_{i=1}^{I} \sum_{j=1}^{s} e_i f_{k_2}(v_{ij})} \bmod p$ and verifies:

$$e(\beta, g) \overset{?}{=} e(\eta, k) \cdot e(y, k) \tag{4}$$

If equation 2 holds, TPS sends *Accpet* to CU to indicate that the data is complete, otherwise sending *Reject* indicates that the data is missing.

What's more, we describe the active relationship between users, TPS and cloud servers in the scheme proposed in Figure 4.

C. SUPPORT FOR DYNAMIC DATA

In dynamic data operation, CU can perform the following five operations according to the size of the data granularity and operation types: inserting data partition (SI), deleting data partition (SD), inserting data block (BI), modifying data block (BM) and deleting data block (BD). The specific process of dynamic operation of data blocks is shown in Figure 5. When CU needs to perform dynamic data operations, it will perform by the following operation procedures:

1) CU generates operation request information $P_{S_1} = (updata, name, i)$ and authorization Au and sends them to TPS. TPS transmits $\{P_S, Au\}$ to CSP, and CSP executes AuthGen verification authorization.
2) CSP calculate the auxiliary information Δ_i and sends the information $P_R = \{\Delta_i, \sigma_R||name\}$ to the TPS as the correspond information.
3) After receiving P_R, TPS uses Δ_i to construct MHT and calculate root node R'', and uses R'' to verify whether $\sigma_R||name$ is valid. If $\sigma_R||name$ is invalid, it sends fail to CSP and CU to indicate that the operation failed. Otherwise, the next step is executed.
4) TPS modifies v_i to v_i^* according to the operation type, and calculates the root node R^* of MHT. TPS calculates the label of data to be updated, generates and sends dynamic data operation request information $P_{R_2} = (BI/BM/BD/SI/SD, name, v_i, \sigma_R^*||name, i, j, m_{ij}^*, \sigma_{ij}^*)$ to CSP.
5) CSP modifies v_i to v_i^* according to P_{R_2}, updates (m_{ij}, σ_{ij}) to $(m_{ij}^*, \sigma_{ij}^*)$, and then modifies $\sigma_R||name$ to $\sigma_R^*||name$. finally, *ProofGen* and *ProofVerify* are executed to verify the integrity of the updated data.

V. THEORETICAL AND PERFORMANCE ANALYSIS

A. CORRECTNESS

In this section, this article introduces the correctness of the system scheme. Due to the reference of MHT and vector v_i, TPS can master the index positions of data blocks and tags in CSP, which can effectively determine whether CSP performs data update honestly. If CSP and TPS can honestly implement the system scheme, the correctness of the integrity audit can be proved by follow equation:

$$\begin{aligned} e(\beta, g) &= e((u^{\sum_{i=1}^{I} \sum_{j=1}^{s} e_i \cdot H_2(name||i||j||v_{ij})} \\ &\quad \cdot g^{\sum_{i=1}^{I} \sum_{j=1}^{s} e_i \cdot (r \cdot m'_{ij} + f_{k_2}(name||i||j||v_{ij}))})^{\varepsilon}, g) \\ &= e(u^{\sum_{i=1}^{I} \sum_{j=1}^{s} e_i \cdot H_2(name||i||j||v_{ij})}, g)^{\varepsilon} \\ &\quad \cdot e(g^{\sum_{i=1}^{I} \sum_{j=1}^{s} e_i \cdot (r \cdot m'_{ij} + f_{k_2}(name||i||j||v_{ij}))}, g)^{\varepsilon} \\ &= e(\eta, k) \cdot e(y, k) \end{aligned} \tag{5}$$

B. SECURITY

In this section, this paper analyzes the security of the system scheme from the following points: the privacy of data,

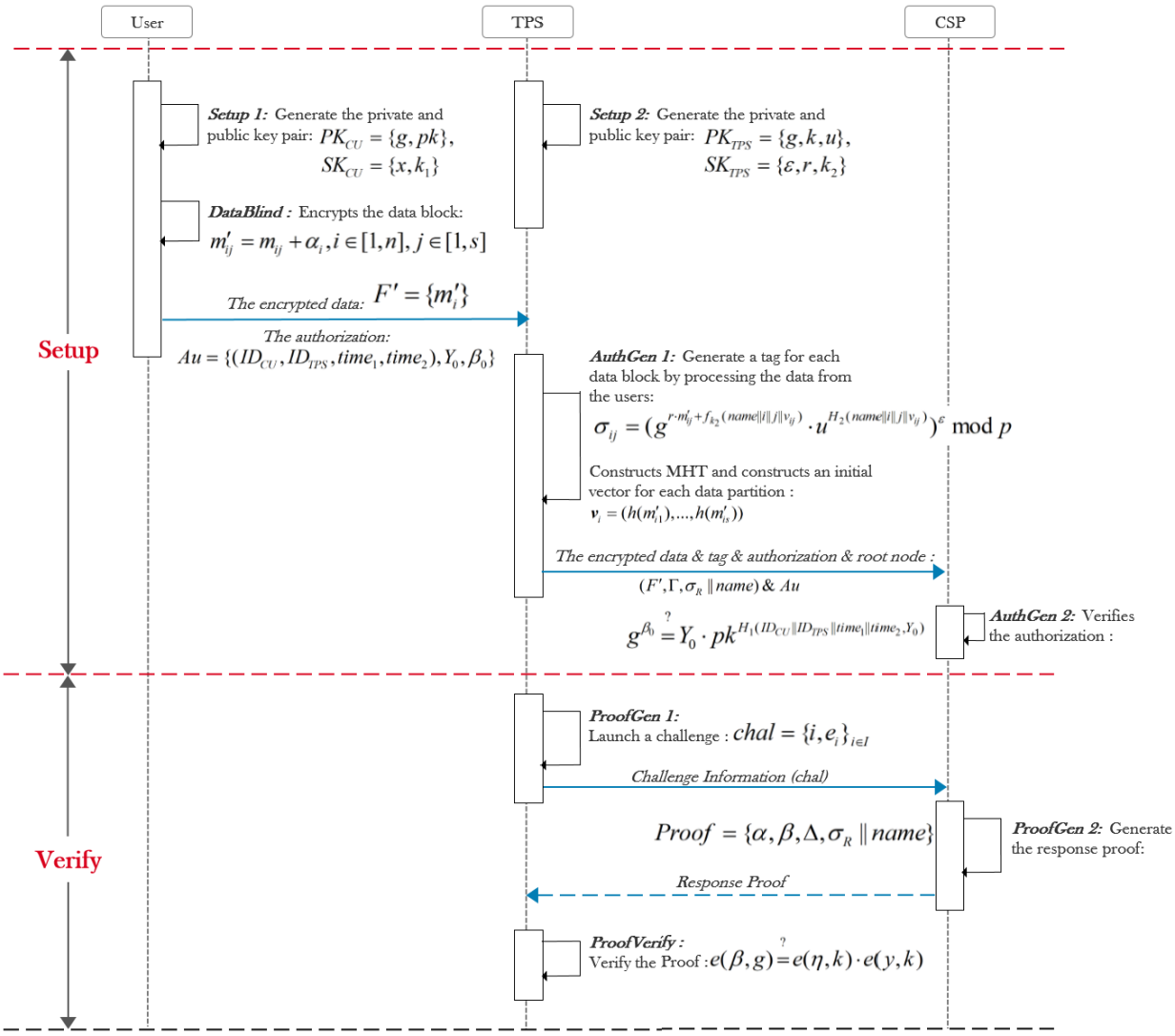


FIGURE 4. Scheme Algorithm Flow Chart.

the inability of the authorization to be forged, and the reliability of audit.

Theorem 1: During the data upload and storage session, TPS and CSP cannot obtain real data by encrypting data.

Proof: Since the encrypted data m'_{ij} is generated by blinding factor α_i, and $\alpha_i = f_{k_1}(i, name)$ is randomly generated by CU through key k_1. Therefore, after receiving the encrypted data m'_{ij}, the TPS cannot obtain the real data m_{ij}. □

Theorem 2: TPS cannot forge authorization without permission and pass CSP inspection. Moreover, TPS cannot operate CSP without permission.

Proof: The unforgeability of the authorization is determined by Y_0 and β_0. However, $Y_0 = g^{r_0}$ is encrypted by CU through private key r_0. Even if TPS knows (g, g^{r_0}), TPS cannot calculate r_0 according to DL theorem. And $\beta_0 = r_0 + x \cdot H_1(ID_{CU}||ID_{TPS}||time_1||time_2, Y_0) \bmod p$ is determined by the CU's private key x, r_0 and the corresponding Y_0. Therefore, TPS cannot forge (Y_0, β_0) to pass CSP's examination in *AuthGen* AuthGen. And through $(ID_{CU}, ID_{TPS}, time_1, time_2)$, TPS can be guaranteed to complete the operation honestly according to CU's instructions. □

Theorem 3: In data integrity audit, CSP cannot cheat TPS through integrity audit by forging, aggregating or using expired data and labels.

Proof: Before auditing the integrity of data, TPS firstly verify the index structure and labels in CSP by the MHT and vector v_i, so that CSP cannot use expired data and labels to pass the audit. According to CDH, CSP cannot calculate the key $\{r, \varepsilon\}$ through the data label, and CSP cannot know the index k_2 of the pseudo-random function $f_{k_2}(\cdot)$ contained in the signature, so the data signature is unforgeable. Moreover, since TPS randomly selects I data blocks for audit, if CSP

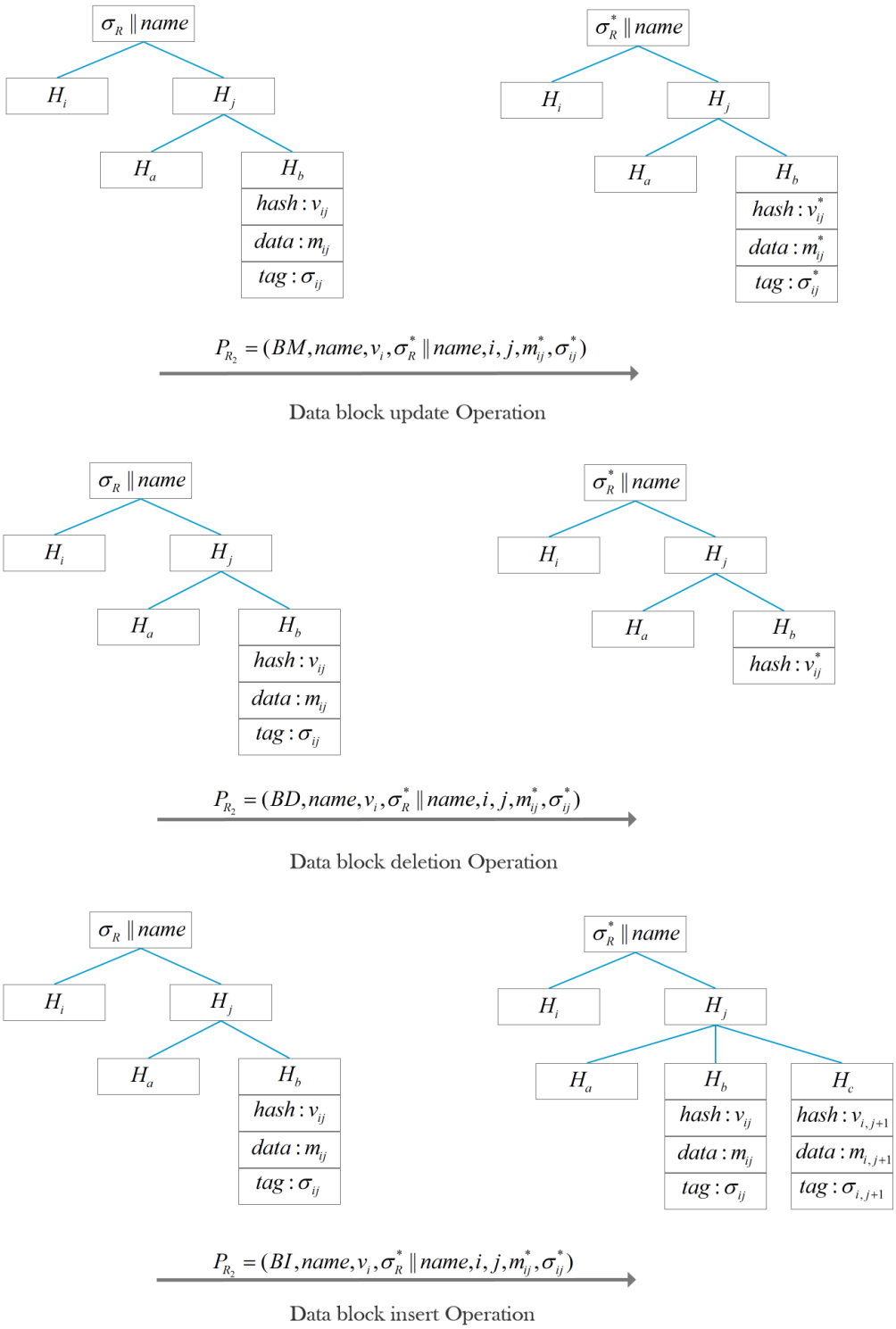


FIGURE 5. Data block operation.

wants to calculate aggregated data blocks in advance, it needs to calculate 2^{I-1} combinations. Therefore, it is unrealistic for CSP to pass the audit by aggregating data and labels. □

Theorem 4: In the data integrity audit, CSP cannot cheat TPS to pass the integrity audit by forging data and labels.

Proof: In order to prove this principle, we designed the following game process (in order to prove it more concisely and effectively, we do not consider MHT path and authorization here): First, TPS sends challenge message $chal = \{i, e_i\}_{i \in I}$ to CSP. In order to pass the verification of

Equation 4, CSP needs to send proof information $Proof = \{\alpha, \beta\}$ to TPS, where $\alpha = \sum_{i=1}^{I}\sum_{j=1}^{s} m'_{ij} \bmod (p-1)$ and $\beta = \prod_{i=1}^{I}\prod_{j=1}^{s} \sigma_{ij}{}^{e_i} \bmod p$. However, due to the damage or missing part of m'_{ij}, CSP cannot calculate the correct α. Next, we will discuss two cases where CSP forges proof information.

Case 1: CSP only forges α', sends $Proof = \{\alpha', \beta\}$ to TPS, and then TPS calculates $y' = g^{r\cdot\alpha' + \sum_{i=1}^{I}\sum_{j=1}^{s} e_i f_{k_2}(v_{ij})} \bmod p$. Assuming that CSP wins this game process, according to Equation 4, we have

$$e(\beta, g) = e(\eta, k) \cdot e(y', k) \tag{6}$$

Furthermore, if $Proof = \{\alpha, \beta\}$ is the correct proof information, then we have

$$e(\beta, g) = e(\eta, k) \cdot e(y, k) \tag{7}$$

According to formula 5 in the correctness proof, we can obtain $g^{\alpha} = g^{\alpha'} \Rightarrow g^{\Delta\alpha} = 1 \Rightarrow \alpha = \alpha'$. That is, unless the CSP guesses the true α value, the CSP cannot deceive the TPS by forging α'. Obviously, the probability of CSP guessing α is $\Pr[Unbound] = negl(\lambda)$, which is almost impossible. Therefore, Case1 does not hold.

Case 2: CSP forges (α', β') at the same time and sends $Proof = \{\alpha', \beta'\}$ to TPS, after which TPS calculates $y' = g^{r\cdot\alpha' + \sum_{i=1}^{I}\sum_{j=1}^{s} e_i f_{k_2}(v_{ij})} \bmod p$. If the CSP wants to win this game process, the CSP must effectively calculate the digital signature corresponding to the damaged data block, namely $\sigma_{ij} = (g^{r\cdot m'_{ij} + f_{k_2}(v_{ij})} \cdot u^{H_2(name||v_{ij})})^{\varepsilon} \bmod p$. However, (r, k_2, ε) is the private key of TPS. According to DL theorem and CDH theorem, it is difficult for CSP to forge signatures. Similarly, CSP cannot forge β' corresponding to α', so Case2 does not hold.

□

C. PERFORMANCE

In this section, this paper specifically analyzes the theoretical performance of our scheme in terms of computational overhead, communication cost and storage cost. Suppose n is the number of data blocks uploaded to the file, l is the length of the audit query, q is the security level, and d is the number of data blocks dynamically operated.

1) COMPUTATION COST

For CU, the participating processes include *Setup* and *DataBlind*. In the whole protocol, Setup and DataBlind steps for the same file are executed only once, and all overhead of these two steps can be shared equally among subsequent data integrity audits, so the computational overhead for CU is $O(1)$.

For TPS, the processes involved include *AuthGen*, *ProofGen*, *ProofVerify* and *Dynamic Data*. The computational overhead of calculating data labels in AuthGen is $O(n)$. The computational overhead of verifying the index structure of the outsourced data and auditing the integrity of the data in the *ProofVerify* and *Dynamic Data* steps is $O(l)$ and $O(d)$, respectively.

For CSP, the processes involved are *AuthGen*, *ProofGen* and *Dynamic Data*. In the AuthGen step, the computational overhead of verifying the TPS authorization is $O(1)$. The computational overhead of calculating the prove information in *ProofGen* and *Dynamic Data* is $O(l)$.

Therefore, The computation cost of each entity in the system scheme is shown in Table 3:

TABLE 3. Computational cost of each entity in each step.

	CU	TPS	CSP
Setup	$O(1)$	-	-
DataBlind	$O(1)$	-	-
AuthGen	-	$O(n)$	$O(1)$
ProofGen	-	$O(1)$	$O(l)$
ProofVerify	-	$O(l)$	-
Dynamic Data	-	$O(d)$	$O(d)$

2) COMMUNICATION COST

In our scheme, for each entity, there are the following steps to carry out communication overhead: 1) CU uploads encrypted files to TPS. 2) CU sends operation request (data integrity verification, dynamic data) to TPS. 3) TPS uploads data to CSP. 4) TPS sends challenge information or operation request to CSP. 5) CSP returns TPS certification information or operation result. 6) TPS returns CU operation result.

Similarly, for the same file, the encrypted data and the communication overhead generated during the upload process can be spread out equally among subsequent operations. Therefore, in the life cycle of the same data file, the communication overhead generated by CU is $O(1)$. For TPS and CSP, in each data integrity audit, the communication overhead depends on the length of the audit query, and in each dynamic data operation, the communication overhead depends on the length of the dynamic operation data block. Therefore, the communication overhead of each entity in the scheme is shown in Table 4:

TABLE 4. Communication cost of each entity.

	CU	TPS	CSP
Communication cost	$O(1)$	$O(l)/O(d)$	$O(l)/O(d)$

3) STORAGE COST

In our solution, CU, CSP and TPS do not need to bear too much storage overhead. For users, only their own private key $SK_{CU} = \{x, k_1\}$ needs to be stored. For the security node TPS, the data information to be saved is mainly its own private key $SK_{TPS} = \{\varepsilon, r, k_2\}$ and public information $PK = \{g, k, u, pk\}$. For CSP, it is mainly responsible for

TABLE 5. Performance Comparison of Different Schemes.

	SCS [12]	DAP [11]	ODA [6]	Our Scheme
Dynamic auditing	*No*	*Yes*	*Yes*	*Yes*
Data structure	-	*ITable*	*RTable*	*MHT*
Data structure management	-	*Auditor*	*TTPA*	*TPS*
Privacy protect	*No*	*No*	*No*	*Yes*
Communication complexity	$O(l)$	$O(l)/O(d)$	$O(l)/O(d)$	$O(1)/O(l)/O(d)$
Server computation complexity	$O(l)$	$O(ls)/O(ds)$	$O(l)/O(d)$	$O(l)/O(d)$
Users computation complexity	$O(l)$	$O(ls)/O(ds)$	$O(d)$	$O(1)$

storing data files and the signature of each data block, but due to the introduction of Merkle Hash tree, its storage overhead will be relatively reduced.

Next, we compare the performance of our scheme with that of other similar schemes recently, and analyze their performance and advantages respectively. The performance pairs of each scheme are shown in Table 5:

We found that the difference between different schemes lies mainly in the structural design of the schemes, which further shows the difference in functions. In SCS protocol, Chen *et al.* [12] implemented digital signature based on distributed string equality checking protocol, which has a good performance in computational efficiency and security. Therefore, in our scheme, we mainly refer to SCS protocol to implement digital signature. In SCS protocol, although it implements simple dynamic data based on hash table, it cannot be applied to practice due to its inefficient operation. Moreover, it does not support public auditing and privacy protection, and users have a huge computing burden. In DAP scheme, Yang and Jia [11] simply implemented data integrity audit based on BLS short signature and bilinear pairing. Although its computational and communication overhead is low, its security is worrying. Based on an index table (ITable), they basically realize all operations of dynamic data, but due to the limitation of data structure, the operation efficiency needs to be improved. In ODA scheme, Gan *et al.* [6] implemented data integrity audit based on algebraic signature and XOR-homomorphic function, and basically realized dynamic data operation through Index Table. However, like DAP scheme, ODA scheme does not protect users' data privacy very well, and users' computing overhead can be further reduced. In our scheme, we use distributed string equality checking protocol to implement data integrity audit by referring to the work of Chen *et al.*, which improves the security of the scheme while maintaining low computation and communication overhead. And through the Merkle Hash tree to achieve dynamic data operations, improve the efficiency of dynamic operations. At the same time, by setting up a secure node TPS, we can ensure the privacy of user data and reduce the local computing overhead of users.

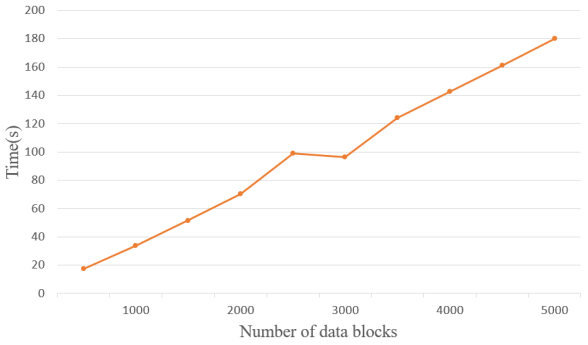


FIGURE 6. Computation cost for different block numbers in digital signature.

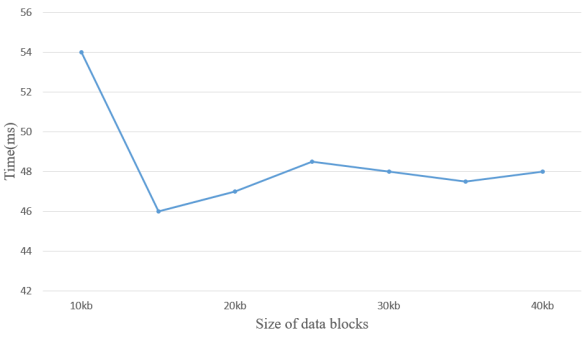


FIGURE 7. Computation cost for different block size in integrity auditing.

We further evaluate the computational cost of our scheme, in the setting of Intel®Core i7-7700HQ CPU @ 2.80GHz/16 GB Ram. According to the Java Pairing-Based Cryptography Library (JPBC), we simulated the computation cost of our scheme under IntelliJ IDEA. We divide each block into 20 sectors. We record the computation time for the digital signature in the AuthGen algorithm where the data blocks are set in 10KB. Then we record the time for integrity auditing in the ProofVerify where the file size is 4MB. All the experimental results are the averages of 20 times. The results of digital signature are shown in Figure 6, and the results of integrity

auditing are shown in Figure 7. From the experimental data, we can see our scheme is efficient for the cloud servers.

VI. CONCLUSION

During the exploration, UAVs continuously upload real-time data to the cloud server. However, the storage and computing capabilities of UAVs are limited, and a public audit scheme supporting dynamic data is needed. More importantly, the privacy of data collected by UAVs must be protected. In this paper, we propose a general and efficient privacy-preserving public cloud audit scheme which supports dynamic data. By designing the third-party server (TPS) and security authorization, we have realized the protection of data privacy and greatly reduced the local computing overhead of UAVs. Based on MHT multi-level index structure, we have realized the dynamic data operation of cloud data, in the meantime, greatly improved the efficiency of the dynamic data operation and the storage efficiency of the cloud server itself. At the same time, we design the digital signature in the scheme based on distributed string equality checking protocol and bilinear mapping. We have verified the safety and theoretical performance of our scheme through detailed linear algebraic derivation and calculation, and we have verified the effectiveness of our scheme through experiments and performance evaluation. The results show that our scheme can effectively realize outsourced data integrity audit. Comparing with the existing schemes, it can effectively reduce computational and storage costs. However, at the same time, we should point out that in order to further optimize the computational efficiency and security of the scheme, we need to consider designing a more optimized signature scheme and a more optimized system structure, which is also our future work.

REFERENCES

[1] 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)*, 2007, pp. 598–609, doi: 10.1145/1315245.1315318.

[2] H. Tian, Y. Chen, C.-C. Chang, H. Jiang, Y. Huang, Y. Chen, and J. Liu, "Dynamic-hash-table based public auditing for secure cloud storage," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 701–714, Sep. 2017, doi: 10.1109/tsc.2015.2512589.

[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, doi: 10.1109/tpds.2010.183.

[4] B. Shao, G. Bian, Y. Wang, S. Su, and C. Guo, "Dynamic data integrity auditing method supporting privacy protection in vehicular cloud environment," *IEEE Access*, vol. 6, pp. 43785–43797, 2018, doi: 10.1109/access.2018.2863270.

[5] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. 16th ACM Conf. Comput. Commun. Secur. (CCS)*, 2009, pp. 213–222, doi: 10.1145/1653662.1653688.

[6] Q. Gan, X. Wang, and X. Fang, "Efficient and secure auditing scheme for outsourced big data with dynamicity in cloud," *Sci. China Inf. Sci.*, vol. 61, no. 12, Dec. 2018, Art. no. 122104, doi: 10.1007/s11432-017-9410-9.

[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, doi: 10.1109/infcom.2010.5462173.

[8] S. G. Worku, C. Xu, J. Zhao, and X. He, "Secure and efficient privacy-preserving public auditing scheme for cloud storage," *Comput. Electr. Eng.*, vol. 40, no. 5, pp. 1703–1713, Jul. 2014, doi: 10.1016/j.compeleceng.2013.10.004.

[9] L. Yang and L. Xia, "An efficient and secure public batch auditing protocol for dynamic cloud storage data," in *Proc. Int. Comput. Symp. (ICS)*, Dec. 2016, pp. 671–675, doi: 10.1109/ics.2016.0138.

[10] M.-Q. Hong, P.-Y. Wang, and W.-B. Zhao, "Homomorphic encryption scheme based on elliptic curve cryptography for privacy protection of cloud computing," in *Proc. IEEE IEEE 2nd Int. Conf. Big Data Secur. Cloud (BigDataSecurity), Int. Conf. High Perform. Smart Comput. (HPSC), IEEE Int. Conf. Intell. Data Secur. (IDS)*, Apr. 2016, pp. 152–157, doi: 10.1109/bigdatasecurity-hpsc-ids.2016.51.

[11] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 9, pp. 1717–1726, Sep. 2013, doi: 10.1109/tpds.2012.278.

[12] F. Chen, T. Xiang, Y. Yang, C. Wang, and S. Zhang, "Secure cloud storage hits distributed string equality checking: More efficient, conceptually simpler, and provably secure," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 2389–2397, doi: 10.1109/infocom.2015.7218627.

[13] D. Brown and R. Gallant, "The static Diffie–Hellman problem," Cryptol. ePrint Arch., Tech. Rep. 2004/306, 2004, p. 306. [Online]. Available: https://eprint.iacr.org/2004/306

[14] 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. (SecureComm)*, 2008, pp. 1–10, doi: 10.1145/1460877.1460889.

[15] A. Laliberte, "Unmanned aerial vehicle-based remote sensing for rangeland assessment, monitoring, and management," *J. Appl. Remote Sens.*, vol. 3, no. 1, Aug. 2009, Art. no. 033542, doi: 10.1117/1.3216822.

[16] S. Siebert and J. Teizer, "Mobile 3D mapping for surveying earthwork using an unmanned aerial vehicle (UAV)," in *Proc. 30th Int. Symp. Automat. Robot. Construct. Mining (ISARC), Building Future Automat. Robot.*, Aug. 2013, pp. 1–9, doi: 10.22260/isarc2013/0154.

[17] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016, doi: 10.1109/twc.2016.2531652.

[18] 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, doi: 10.1109/tc.2014.2375190.

[19] L. Rao, H. Zhang, and T. Tu, "Dynamic outsourced auditing services for cloud storage based on batch-leaves-authenticated Merkle hash tree," *IEEE Trans. Services Comput.*, early access, May 26, 2017, doi: 10.1109/tsc.2017.2708116.

[20] P. M. Pardeshi and D. R. Borade, "Enhancing data dynamics and storage security for cloud computing using merkle hash tree and AES algorithms," *Int. J. Comput. Appl.*, vol. 98, no. 21, pp. 1–7, Jul. 2014, doi: 10.5120/17304-7734.

[21] J. Zhang, S. Liu, and J. Mao, "On the security of a dynamic-hash-table based public auditing protocol," in *Proc. 9th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2016, pp. 1954-1958, doi: 10.1109/cisp-bmei.2016.7853038.

[22] Y. Wang, J. Li, and H. H. Wang, "Cluster and cloud computing framework for scientific metrology in flow control," *Cluster Comput.*, vol. 22, no. S1, pp. 1189–1198, Jan. 2019, doi: 10.1007/s10586-017-1199-3.

[23] B. Hayes, "Cloud computing," *Commun. ACM*, vol. 51, no. 7, pp. 9–11, Jul. 2008, doi: 10.1145/1364782.1364786.

[24] Y. Yan, L. Wu, G. Gao, H. Wang, and W. Xu, "A dynamic integrity verification scheme of cloud storage data based on lattice and Bloom filter," *J. Inf. Secur. Appl.*, vol. 39, pp. 10–18, Apr. 2018.

[25] G. S. Aujla, R. Chaudhary, N. Kumar, A. K. Das, and J. J. P. C. Rodrigues, "SecSVA: Secure storage, verification, and auditing of big data in the cloud environment," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 78–85, Jan. 2018.

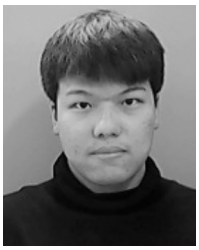
[26] A. Fu, Y. Li, S. Yu, Y. Yu, and G. Zhang, "DIPOR: An IDA-based dynamic proof of retrievability scheme for cloud storage systems," *J. Netw. Comput. Appl.*, vol. 104, pp. 97–106, Feb. 2018.

[27] W. Xu, L. Wu, and Y. Yan, "Privacy-preserving scheme of electronic health records based on blockchain and homomorphic encryption," *J. Comput. Res. Develop.*, vol. 55, no. 10, pp. 2233–2243, 2018.

[28] Z. Yang, W. Wang, Y. Huang, and X. Li, "Privacy-preserving public auditing scheme for data confidentiality and accountability in cloud storage," *Chin. J. Electron.*, vol. 28, no. 1, pp. 179–187, Jan. 2019.

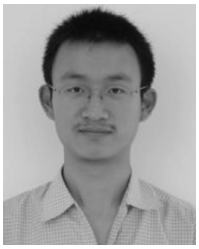
[29] N. Garg and S. Bawa, "RITS-MHT: Relative indexed and time stamped merkle hash tree based data auditing protocol for cloud computing," *J. Netw. Comput. Appl.*, vol. 84, pp. 1–13, Apr. 2017.

[30] D. Kim, J. Son, D. Seo, Y. Kim, H. Kim, and J. T. Seo, "A novel transparent and auditable fog-assisted cloud storage with compensation mechanism," *Tsinghua Sci. Technol.*, vol. 25, no. 1, pp. 28–43, Feb. 2020.
[31] Y. Lin, J. Li, X. Jia, and K. Ren, "Multiple-replica integrity auditing schemes for cloud data storage," *Concurrency Comput., Pract. Exper.*, vol. 8, Aug. 2019, Art. no. e5356, doi: 10.1002/cpe.5356.
[32] D. Liu, J. Shen, P. Vijayakumar, A. Wang, and T. Zhou, "Efficient data integrity auditing with corrupted data recovery for edge computing in enterprise multimedia security," *Multimedia Tools Appl.*, vol. 79, nos. 15–16, pp. 10851–10870, Apr. 2020.
[33] S. K. Nayak and S. Tripathy, "SEPDP: Secure and efficient privacy preserving provable data possession in cloud storage," *IEEE Trans. Services Comput.*, early access, Mar. 29, 2018, doi: 10.1109/TSC.2018.2820713.
[34] S. Rass, "Dynamic proofs of retrievability from Chameleon-Hashes," in *Proc. 10th Int. Conf. Secur. Cryptogr.*, 2013, pp. 1–9.
[35] S. Sondur and K. Kant, "Towards automated configuration of cloud storage gateways: A data driven approach," in *Proc. 12th Int. Conf., Held Services Conf. Fed. (SCF)*, San Diego, CA, USA, Jun. 2019, pp. 192–207.
[36] E. Torres, G. Callou, and E. Andrade, "A hierarchical approach for availability and performance analysis of private cloud storage services," *Computing*, vol. 100, no. 6, pp. 621–644, Jun. 2018.



JIASEN LIU is currently pursuing the master's degree with the Engineering University of the People's Armed Police. His main research interests include public key cryptography and cloud security.

XU AN WANG is currently a Professor with the Engineering University of the People's Armed Police. He has published about 100 articles in the field of information security. His main research interests include public key cryptography and cloud security.



ZHIQUAN LIU received the B.S. degree from the School of Science, Xidian University, in 2012, and the Ph.D. degree from the School of Computer Science and Technology, Xidian University, in 2017. He is currently a Lecturer with the College of Information Science and Technology, Jinan University. His current research interests include trust modeling and privacy protection in the Internet of Vehicles.

HAN WANG received the B.S. degree in computer science and technology from Northeastern University, Shenyang, China, in 2018. He is currently pursuing the M.S. degree in cryptography with the Engineering University of the People's Armed Police, Xi'an, China. His main research interests include information security and cloud storage.

XIAOYUAN YANG is currently a Professor with the Engineering University of the People's Armed Police. He has published about 100 articles in the field of information security. His main research interest includes cryptography.