IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 14, 2020, accepted January 29, 2020, date of publication February 12, 2020, date of current version February 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2973354

A Data Storage and Sharing Scheme for Cyber-Physical-Social Systems

LONGXIA HUANG[1], GONGXUAN ZHANG[2], (Senior Member, IEEE), AND SHUI YU[3], (Senior Member, IEEE)

[1]Department of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China
[2]School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[3]School of Software, University of Technology Sydney, Broadway, NSW 2007, Australia

Corresponding author: Longxia Huang (hlongxia2017@163.com)

This work was supported in part by the National Science Foundation of China under Grant U1736216, Grant 61802185, and Grant U1836116, and in part by the Natural Science Foundation of Jiangsu Province under Grant BK20180470.

ABSTRACT Cyber-Physical-Social System (CPSS) provides users secure and high-quality mobile service applications to share and exchange data in the cyberspace and physical world. With the explosive growth of data, it is necessary to introduce cloud storage service, which allows devices frequently resort to the cloud for data storage and sharing, into CPSS. In this paper, we propose a data storage and sharing scheme for CPSS with the help of cloud storage service. Since data integrity assurance is an inevitable problem in cloud storage, we first design a secure and efficient data storage scheme based on the technology of public auditing and bilinear map, which also ensures the security of the verification. In order to meet the real-time and reliability requirements of the CPSS, the rewards of timeliness incentive and effectiveness incentive are considered in the scheme. Secondly, based on the proposed storage scheme and ElGamal encryption, we propose a lightweight access model for users to access the final data processed by cloud server. We formally prove the security of the proposed scheme, and conduct performance evaluation to validate its high efficiency. The experimental results show that the proposed scheme has lower overheads in communication and access as compared to the technique CDS.

INDEX TERMS Data sharing, integrity checking, access control, cyber-physical-social systems, cloud computing.

I. INTRODUCTION

Cyber-Physical-Social Systems (CPSS) [1] integrate the cyber, physical and social spaces together to provide users with a convenient and intelligent environment [2], [3]. The CPSS allow a large number of social users to have more interaction than before by data sharing and exchanging, which results in the explosive growth of the data generated and collected from social and physical spaces [4]. Generally, these data has the properties of 4Vs: volume, variety, velocity, and veracity [5]. Therefore, how to store amd process the data generated in CPSS becomes a key issue.

Cloud storage service makes it convenient for the devices to frequently store and share the outsourced data, which makes up for the defects of CPSS but also would bring a series of security challenges inevitably [6]. Firstly, the integrity of the outsourced files should be guarenteed since the data stored in the cloud may be corrupted due to hardware failures, manual operation errors, or external attack. Beside integrity checking, the efficient is also a research point should be concerned, so we introduce the technique of public auditing to ensure the integrity of outsourced data. Better than the traditional integrity checking scheme, public auditing makes the verifier to check the integrity without downloading all files with the help of a challenge-and-response protocol which can realize sampling inspection with a high detection rate. What's more, the integrity checking by a third public party greatly reduces the burden of file owners on computation resource and it is on-line all the time. Last but not least, privacy protection cannot be ignored in public auditing since the public verifiers are curious about the data privacy during this verification [7]. As we introduce cloud storage service in CPSS, the above requirements such as integrity checking, efficiency and privacy protection should all be satisfied in this paper.

To further improve the performance of data collection in CPSS, the incentive mechanism to provide efficient crowd sourcing becomes a challenge. The crowd sourcing plays an important role as it encourages social users to provide data to

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

the CPSS rather than only receive data from others [8]. The various data collected by users in the CPSS can be further processed for sensing, monitoring, and communicating with the cyberspace and physical world [9]. For example, when a new mobile user joins the crowd sourcing in the CPSS, he/she can utilize the sensors equipped in his/her mobile sevice to provide the information around. These information contains traffic, weather, event information and so on. The CPSS can collect these information to monitor the environment and further process it to generate the final data for users to access. In this paper, we set two kinds of incentive: the timeliness reward and effectiveness reward to encourage users to join the data collection. The setting also ensures the real-time [10], [11] and reliability [12] in CPSS.

The property of lightweight is another important issue, so the access method should be low-overhead [2]. To realize lightweight, the attribute can be transformed to a numeric parameter by weighted attribute [13]. The confidentiality and availability are ensured by access control to prevent unauthored users from file obtaining [13]. We use weighted attribute [14] to optimize the access structure and ElGamal encryption [15] to protect the security of the access key while reduce the overhead of access respectively.

In this paper, we aim to design an efficient and secure data storage and sharing scheme for Cyber-Physical-Social Systems. In our scheme, users are encouraged to take part in the data collection with rewards, such that the timeliness and effectiveness can be realized. The privacy of the stored data is preserved by using the pseudo-random method during the integrity verification of files stored in the cloud and the proposed access method meets the requirement of lightweight. The major contributions of this paper are summarized as follows.

1) We propose an secure and efficient data storage scheme for CPSS based on the technology of public auditing. The proposed data storage scheme can achieve secure and efficient auditing since it utilizes pseudo-random permutation and pseudo-random function to generate the challenge message.
2) To meet the requirements of CPSS such as real-time and reliability, we set two kinds of incentive value: the timeliness incentive and effectiveness incentive in this paper to encourage users to take part in the data collection.
3) We deploy an efficient access method based on ElGamal encryption in this paper, which significantly reduces the overhead of file access for users. The improved method meets the lightweight requirements.
4) We analyze the security in this paper and carry out extensive experiments to evaluate the performance of the proposed scheme in reducing the overhead of computation and communication. Simulation results demonstrate that the proposed scheme outperforms the compared scheme: Customized Data Sharing Scheme Based on Blockchain and Weighted Attribute (CDS) in access.

Organization: The rest of this paper is organized as follows. The related work about the proposed scheme is summarized in Section II. Section III lists some preliminaries for understanding the proposed scheme easily. The design goals and models are described in Section IV. The details of the data storage and sharing scheme are shown in Section V. The analysis of the proposed scheme is presented in Section VI and Section VII shows the experimental results. Finally, we conclude this paper and list some future work in Section VIII.

II. RELATED WORK

As the dishonest behavior of the cloud server and the occurrence of software errors threaten the data integrity, Ateniese *et al.* [16] proposed the definition of provable data possession, which is the base of integrity checking schemes. Zhang *et al.* [17] utilized public auditing by delegating the auditing task to a third party to handle the problem of users' limited computation source. Public auditing schemes must pay attention to the leakage of data privacy [5] as the public verifier is naturally curious about the sensitive information on the files and will attempt to get private information from the received message. To solve the problem, Yu *et al.* [18] introduced ''zero-knowledge privacy" to realize privacy protection by ensuring that the verifier cannot learn any knowledge about the stored data during the efficient auditing process. However, the above public auditing methods are not efficient due to their high communication and computation cost. In this paper, we utilize ElGamal encryption [15] to achieve privacy protection with efficiency.

In data sharing schemes, the free sharing may attack the enthusiasm of users and result in the low motivation of crowd sensing [9]. The concept of incentive was introduced in public auditing schemes by Wang et al. [19] to encourage users to disclose bad issues. An incentive data sharing scheme CDS [14] utilized the technology of blockchain to encourage users to share data and support customization based on attribute-based encryption (ABE) [20]. Su *et al.* [8] designed an optimal method by dividing social users into three types to select social users to join crowd sourcing in CPSS.

The important cryptography primitive in access control is ABE. Different from the fuzzy identity-based encryption (IBE) [21], the basic technology, ABE provides a one-to-many file sharing by setting a access structure which limits the visitors. As traditional ABE scheme [20] cannot support attribute with an arbitrary state, scheme [13] introduced weighted ABE to show the importance of different attributes in the same set. The conception of weighted attribute optimizes the access structure and reduces overhead by storing attribute values instead of attribute strings.

III. PRELIMINARIES

Before detailing the proposed scheme, we introduce the technology of ElGamal encryption and Bilinear map, which are used in this paper.

A. ELGAMAL ENCRYPTION

In this paper, the security and availability of the proposed scheme is based on ElGamal encryption. T. ElGamal [15] proposed a digital signature method ElGamal in 1985. The method is based on the public key cryptosystem of discrete logarithm problem in finite field. An ElGamal encryption has the following steps.

1) Select a prime p, a prime field $Z_p = \{1, 2, \cdots, p-1\}$ and a generator of Z_p as g.
2) Choose a random number $x \in [1, p-1]$ to calculate $y = g^x \pmod p$.
3) Choose a random number $k \in [1, p-1]$ to encrypt message m as $C = (C_1, C_2)$ where $C_1 = g^k \pmod p$ and $C_2 = my^k \pmod p$.
4) Decrypt $C = (C_1, C_2)$ by computing $m = C_2(C_1^x)^{-1} \pmod p$.

The secret key is x and the public key is $\{y, g, p\}$. It's hard to calculate the secret key x with the public information $\{y, g, p\}$ [22]. Therefore, we use the asymmetric encryption ElGamal encryption to enable secure key transfer.

B. BILINEAR MAP AND SECURITY ASSUMPTION

The integrity checking of stored files is based on the properties of bilinear map and the security proof is under Computational Diffie-Hellman assumption in random oracle model.

Definition 1 (Bilinear Map [23]): Let G_1 and G_2 be two multiplicative cyclic groups of prime order q, and g be a generator of group G_1. Then a bilinear map is $e : G_1 \times G_1 \rightarrow G_2$, which has the following properties.

1) For any elements $x, y \in G_1$ and any primes $a, b \in Z_q$, the equation $e(x^a, y^b) = e(x, y)^{ab}$ holds, where Z_q is the set of prime numbers.
2) For any elements $x_1, x_2, v \in G_1$, the equation $e(x_1 \cdot x_2, v) = e(x_1, v) \cdot e(x_2, v)$ holds.
3) There must be a method to calculate the value of map e efficiently, which is non-trivial as $e(g, g) \neq 1$.

C. HOMOMORPHIC AUTHENTICATION

Homomorphic authentication is the basis for integrity checking by enabling an auditor to verify the integrity of a file without downloading the full file. It has three properties: unforgeability, blockless verification and non-malleability.

- Unforgeability: a party without the correct private key is unable to generate a valid message.
- Blockless verification: an auditor verifies the correctness of file with a linear combination of blocks without knowing the details of the file.
- Non-malleability: a party without the valid secret keys cannot use the given signatures to generate a valid signature on the combined blocks.

IV. DESIGN GOALS AND SYSTEM MODELS

In this section, we talk about the design goals and system models of the proposed data storage and sharing method.

A. DESIGN GOALS

The proposed method has the properties as follows. **1) Integrity.** The proposed scheme allows each user to collect data and store it in the cloud server who is asked to prove the integrity of the stored data due to same software errors. **2) Efficiency and security.** What's more, the communication overhead is low and the integrity checking is secure against the public verifier in the proposed scheme. **3) Unique access.** For the contribution of data owners, they are rewarded with value of timeliness incentive and effectiveness incentive which is used to get the access of the final processed file. The access is also lightweighted. To meet the requirements of CPSS, we propose a lightweighted file access method.

B. SYSTEM MODELS

The proposed system in this paper consists of three models: the data storage model, the public verification model and the access model. In the data storage model, each user collects the data, signs the data and uploads it with its signature to the cloud server. Once receives the message from the users, the cloud server timestamps the data and stores the set of the data information. The user is allowed to check the integrity of the stored data information with the help of the public verification model. The major technology of the public verification model is a challenge-and-responce protocol. The access model gives the users who meet the requirement of access threshold the rights to access the final data. The details of these three models are as follows.

1) DATA STORAGE MODEL

The data storage model helps users to generate signature on their collected data and further be used to check data integrity. After getting the data from users, the cloud server timestamps it to store. As shown in Fig. 1, the data storage model is composed of three entities: 1) CA, a certificate authority (CA) who generates key pairs and the public parameters; 2) users, who collect data, sign it and store it in the cloud server; 3) the cloud server, which has sufficient storage capacity and computation resources to process files.

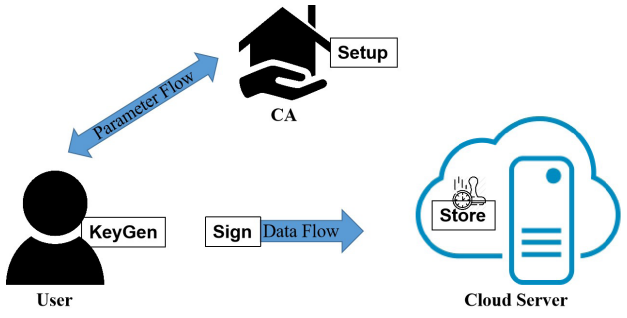

FIGURE 1. The model of data store.

CA first runs algorithm **Setup** to generates the public parameters for the system. Then, the user runs algorithm **KeyGen** to get the key pair and further use algorithm **Sign** to

generate the signature of the collected data. Finally, the data and its signature are timestamped by the cloud server by algorithm **Store** before storing.

2) PUBLIC VERIFICATION MODEL

As shown in Fig. 2, the public verification model is composed of three entities: 1) users, who have a large quantity of collect files to be stored in the cloud or have the right to download these files; 2) a public verifier, who is delegated by the honest-but-curious user to verify the integrity of stored data by a challenge-and-response protocol, since he/she may be curious about the data information; 3) the cloud server, which has sufficient storage capacity and computation resources to process files but may inadvertently modify or delete the original stored data due to hardware failures or manual operation errors. To maintain its reputation, the cloud may conceal the fact that the data is damaged [5]. Therefore, the cloud is not fully trusted and should be checked.

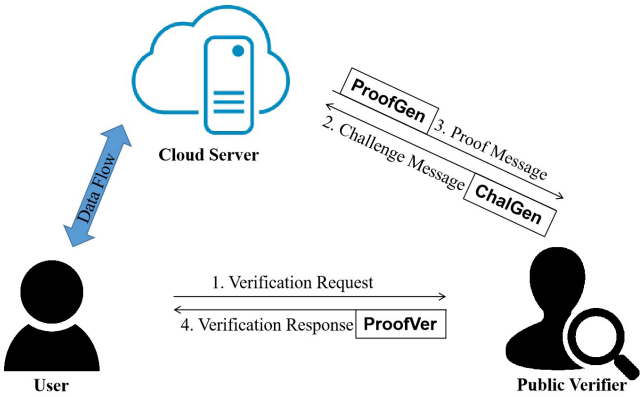

FIGURE 2. The public verification model.

Public verification can greatly reduce users' burden on checking the integrity of the stored data and also protects data privacy during the checking. In general, public auditing operates as follows. Firstly, a user sends a checking request to the public verifier, which immediately sends an auditing challenge message to the cloud server by running **ChalGen** once receiving the request. Based on the stored data, the cloud server will reply an auditing proof of data possession to the public verifier after receiving the challenge with the help of **ProofGen**. Finally, the public verifier runs **ProofVer** to verify the correctness of the proof and then returns the result back to the user.

3) ACCESS MODEL

After data collection, the cloud server will process the data collected by all users to generate the final shared data. For encouragement, each user who contributes the final data will get incentive and the incentive value is set by the cloud server according to the value of the collected data. Noting that there are two kinds of incentive: the timeliness incentive given in **Store** and effectiveness of the collect data in this part. The access model is composed of two entities: user and the cloud server.

In data access, as shown in Fig. 3, the cloud server first sets the paramaters including the access threshold and the weights of two kinds of incentive through **VSet**. Then, the cloud server runs **ASet** to encrypt the availability incentive of the user. With the help of **GetA**, the user gets the attribute values with the secret key. Finally, the user accesses the final data with the help of algorithm **Access** if the value of his/her attribute set is bigger than the access threshold.

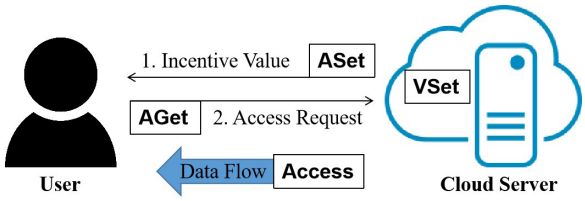

FIGURE 3. The proposed access model.

V. THE PROPOSED SCHEME

The proposed data storage and sharing scheme consists of three parts: data collection, data checking and data access. In data collection, each member collects information and signs the information with his/her secret key before uploading the pair of information and the corresponding signature to the cloud server. After getting the message from a user, the cloud server sets the values of each data according to the timeliness and effectiveness of the data and encrypts the values with the public key of the data owner. To ensure the collected data is correctly stored in the cloud, the data owner is allowed to check the integrity of the stored data. With the secret key, the data owner is able to get the encrypted value and accesses the final data if the value meets the access conditions in data access. The details of data prosessing are not considered in this paper.

To describe the proposed scheme clearly, we show the meanings of the notations in TABLE 1 first.

A. THE DETAILS OF DATA COLLECTION

Every user is able to collect data for data processing. The details of data collection consist of four algorithms: **Setup**, **KeyGen**, **Sign** and **Store**.

Setup: This algorithm is run by CA to generate the public system parameter set $(g, G_1, G_2, q, H, f, \pi)$. Given a secret parameter s, this algorithm chooses two multiplicative cycle groups G_1 and G_2 of prime order q respectively where q is a large prime number. Then it selects a generator g from the multiplicative cycle groups G_1, a hash fuction $H : Z_q \rightarrow G_1$ and a bilinear map $e : G_1 \times G_1 \rightarrow G_2$. Choose a pseudo-random function f as $f : Z_q \times \{1, 2, \cdots, n\} \rightarrow Z_q$ and a pseudo-random permutation $\pi : Z_q \times \{1, 2, \cdots, n\} \rightarrow \{1, 2, \cdots, n\}$, respectively. Finally, it outputs the public parameter set $(g, G_1, G_2, q, H, f, \pi)$.

TABLE 1. Meaning of Notations.

Notation	Meaning	Notation	Meaning	Notation	Meaning
π	A pseudo-random permutation	f	A pseudo-random function	σ_i	The signature of data m_i
$w_{AI_{FD}}$	The availability incentive weight	φ	The aggregated block tags	at_{FD}	The access threshold
$w_{TI_{FD}}$	The timeliness incentive weight	$\{A_{a,i}\}_{i\in[1,n]}$	The availability incentive set	$S'_{a,i}$	The stored file set
(C_1, C_2)	The encrypted incentive	av_a	The access value of user u_a	ak_{FD}	the access key of file FD
Mul_{G_1}	A multiplication operation in G_1	Exp_{G_1}	An exponentiation operation in G_1	$Hash$	A Hash operation

KeyGen: The user u_a with identity ID_a runs this algorithm to generate his/her key pairs as (usk_a, upk_a). The user selects a random number $x_a \in Z_q$ as his/her secret key $usk_a = x_a$ and calculates his/her public key as $upk_a = g^{x_a} \in G_1$.

Sign: This algorithm is run by a data owner u_a to generate the signature σ_i of data m_i with file block tag FID_i before uploading. With the user secret key usk_a, the data message m_i and its identity FID_i, the signature σ_i is calculated as $\sigma_i = (H(FID_i) \cdot g^{m_i})^{usk_a} \in G_1$. Finally, the user uploads the set $S_{a,i} = \{upk_a, FID_i, m_i, \sigma_i\}$ to the cloud server.

Store: The cloud server runs this algorithm to store the message. After getting the set $S_{a,i} = \{upk_a, FID_i, m_i, \sigma_i\}$, the cloud server adds a timestamp to the set and stores the message collected by user u_a as $S'_{a,i} = \{upk_a, FID_i, m_i, \sigma_i, T_{a,i}\}$, where the timestamp $T_{a,i}$ is used to incentive user u_a.

B. THE DETAILS OF DATA CHECKING

In data checking, the data owner checks the integrity of the collected files of user u_a by a challenge-and-response protocol with the help of a public verifier through three algorithms: **ChalGen**, **ProfGen** and **ProfVer**. For ease of description, we assume the files collected by user u_a as $F = \{m_i\}_{1\le i\le n}$, where n is a large number and denotes the number of the file blocks m_i.

ChalGen: After getting the verify request from the data owner u_a, the public verifier generates the challenge message $chal = \{c, k_1, k_2\}$ in this algorithm. The public verifier generates a random set named challenge message $chal = (c, k_1, k_2)$ by choosing two random numbers $k_1, k_2 \in Z_q$ and the number of challenge file blocks c.

ProofGen: Once receivng the challenge message from the public verifier, the cloud server generates the proof message $Pro = \{\{FID_\iota\}_{\iota\in I}, \mu, \varphi\}$ in this algorithm. With the challenge message $chal = (c, k_1, k_2), 1 \le c \le n$, based on the information of the stored set of user u_a as $S'_{a,i} = \{upk_a, FID_i, m_i, \sigma_i, T_{a,i}\}, 1 \le i \le n$, the cloud server first computes $\iota = \pi_{k_1}(j) \in [1, n], v_\iota = f_{k_2}(j) \in Z_q$ for each $j \in [1, c]$. Then, the cloud server aggregates the block tags as $\varphi = \Pi_{\iota\in I}\sigma_\iota^{v_\iota} \in G_1$ where I is the set of ι. The cloud server also needs to calculate $\mu = \sum_{\iota\in I} v_\iota m_\iota \pmod{p-1}$, and sends audit proof $Pro = \{\{FID_\iota\}_{\iota\in I}, \mu, \varphi\}$ to the public verifier.

ProofVer: With the proof message, the public verifier is able to verifiy the file integrity with the help of this algorithm. The public verifier computes

$$\iota = \pi_{k_1}(j), v_\iota = f_{k_2}(j), 1 \le j \le c$$

and then checks the proof by

$$e(\varphi, g) = e(\Pi_{\iota\in I} H(FID_\iota)^{v_\iota} \cdot g^{\mu}, upk_a). \quad (1)$$

If (1) holds, the file F stored in the cloud is deemed as valid and the public verifier returns the result '0' to the cloud server.

C. THE DETAILS OF DATA ACCESS

After data collection, the cloud server will process the data to generate the final shared data and each user who contributes to the final data will get incentive. The incentive is reflected in two aspects: the timeliness and effectiveness of the collect data. The details of data processing is ignored in this paper, so we just regard the timeliness and the availability of the stored data set $S'_{a,i}$ as $T_{a,i}$ and $A_{a,i}$ respectively. Noting that the owner of the set is user u_a and he/she is able to decrypt the encrypted availability incentive $En_{upk_a}(A_{a,i})$ with his/her secret key as the cloud server encrypts the incentive value with the public key of user u_a.

In data access, the cloud server first sets the paramaters including the access threshold and the weights of two kinds of incentive through **VSet**. Then, in **ASet**, the cloud server encrypts the availability incentive of the user. The user u_a gets the attribute values by algorithm **GetA**. When the value of his/her attribute set is larger than the access threshold, the user is allowed to access the final data with the help of algorithm **Access**. Otherwise, the user needs to buy some attribute values to reach the access threshold.

VSet: The cloud server runs this algorithm to generate the set $\{at_{FD}, w_{TI_{FD}}, w_{AI_{FD}}\}$. According to all availability incentive of file FD, the cloud server sets the access threshold as at_{FD}, the weight of the timeliness incentive $w_{TI_{FD}}$ and the weight of the availability incentive $w_{AI_{FD}}$.

ASet: This algorithm is run by the cloud server to encrypts the availability incentive of the user. Given the availability incentive set of user u_a as $\{A_{a,i}\}_{i\in[1,n]}$, the cloud server links all values $A_{a,i}$ in file order to form attribute chain values as $A_a = a_1||a_2||\cdots||a_n$. The cloud server selects a random number $r \in Z_q$ to calculate $C_1 = g^r \in G_1$ and $C_2 = upk_a^r \cdot A_a$. Finally, the cloud server sends the encrypted availability incentive (C_1, C_2) to user u_a.

GetA: With the user key pair (usk_a, upk_a) of user u_a and the encrypted availability incentive $En_{upk_a}(A_a) = (C_1, C_2)$, the user first decrypts it as $A_a = C_2 \cdot (C_1^{usk_a})^{-1}$ to further get the availability incentive $A_{a,i}$. The user u_a gets the timeliness incentive $T_{a,i}$ by accessing the stored data set. Finally, the user u_a gets the value of the attribute set $AS_i = (TI_i, AI_i)$ as $as_{a,i} = (T_{a,i}, A_{a,i})$

Access: User u_a runs this algorithm to access the final data FD with access key ak_{FD}. With the help of algorithm **GetA**, the user u_a gets all the value of the attribute set as $as_a = \{as_{a,i}\}_{i\in[1,n]}$. Assume that the weight of the timeliness incentive and the availability incentive of the final data FD are $w_{TI_{FD}}$ and $w_{AI_{FD}}$ respectively, the access threshold is at_{FD} and the value set $as_a = \{as_{a,i}\}_{i\in[1,n]}$, the cloud server calculates the access value as $av_a = w_{TI_{FD}} \sum_{i\in I} T_{a,i} + w_{AI_{FD}} \sum_{i\in I} A_{a,i}$. If av_a is not less than the access threshold at_{FD}, the cloud server gives the access key ak_{FD} to user u_a.

VI. SECURITY ANALYSIS

This section provides a detailed analysis on the security of the proposed scheme from four perspectives: homomorphic authentication and integrity to ensure public auditing, data privacy against public verifiers, and availability for access.

Theorem 1 (Homomorphic Authentication): The proposed scheme supports homomorphic authentication.

Proof: Homomorphic authentication meets the demands of blockless verifiability and non-malleability, which is the basic tool to construct public verification mechanism. Therefore, we need to prove that the proposed scheme supports blockless verifiability and non-malleability [24]. Given two blocks m_1 and m_2 with the user public key upk, the identifiers FID_1 and FID_2, the signatures σ_1 and σ_2, and two random numbers y_1 and y_2, a verifier is required to check the correctness of the combined block $m' = y_1m_1 + y_2m_2$ by checking the following equation:

$$e\left(\prod_{i=1}^{2}\sigma_i^{y_i}, g\right) = e\left(\prod_{i=1}^{2}\left(H(id_i)^{y_i}\cdot g_1^{m'}\right), pk\right). \tag{2}$$

It's clear that the checking does not need to know in advance the value of blocks m_1 and m_2. Based on the properties of bilinear maps, the proof of (2) is described by (3).

$$\begin{aligned} e\left(\prod_{i=1}^{2}\sigma_i^{y_i}, g\right) &= e\left(\prod_{i=1}^{2}\left(H(id_i)\cdot g_1^{m_i}\right)^{\alpha\cdot y_i}, g\right) \\ &= e\left(\prod_{i=1}^{2}\left(H(id_i)\cdot g_1^{m_i}\right)^{y_i}, g^{\alpha}\right) \\ &= e\left(\prod_{i=1}^{2}\left(H(id_i)^{y_i}\cdot g_1^{m'}\right), pk\right). \end{aligned} \tag{3}$$

Therefore, the proposed scheme supports blockless verification. We then prove the second property that an attacker without knowing the private key cannot generate a valid signature σ' for the combined block $m' = y_1m_1 + y_2m_2$ by combining σ_1 and σ_2 with y_1 and y_2. As hash function H is a one-way hash function, it's impossible for the attacker to success. Let $\theta = [H(FID')g^{m'}]^{\alpha}$ denote the correct signature of block m'. With $\sigma_1^{y_1}\cdot\sigma_2^{y_2} = [\prod_{i=1}^{2}(H(FID_i)^{y_i}\cdot g_1^{m'})]^{\alpha}$, if θ can pass the verification, apparently we have $\prod_{i=1}^{2}H(FID_i)^{y_i} = H(FID')$, which contradicts to the assumption that H is a one-way hash function.

Therefore, the proposed scheme is a homomorphic authenticable scheme. □

Theorem 2 (Integrity): The cloud server is able to pass the integrity verification if it indeed stores the right files and follows the challenge-and-response protocol to generate the proof message honestly.

Proof: In algorithm **ProofGen**, after receiving the challenge message $chall = (c, k_1, k_2), 1 \le c \le n$, the cloud server first computes $\iota = \pi_{k_1}(j) \in [1, n], v_\iota = f_{k_2}(j) \in Z_q$ for each $j \in [1, c]$ based on the challenge message. With the information of the stored set of user u_a as $S'_{a,i} = \{upk_a, FID_i, m_i, \sigma_i, T_{a,i}\}, 1 \le i \le n$, the cloud server calculates the block tags as $\varphi = \Pi_{\iota\in I}\sigma_\iota^{v_\iota} \in G_1$ where I is the set of ι to generate the proof message $Pro = \{\{FID_\iota\}_{\iota\in I}, \mu, \varphi\}$. Then, based on the property of bilinear map in Section 1 we have (4) as follows.

$$\begin{aligned} e(\varphi, g) &= e(\Pi_{\iota\in I}\sigma_\iota^{v_\iota}, g) \\ &= e(\Pi_{\iota\in I}((H(FID_\iota)\cdot g^{m_\iota})^{usk_a})^{v_\iota}, g) \\ &= e(\Pi_{\iota\in I}(H(FID_\iota)\cdot g^{m_\iota})^{v_\iota}, g^{usk_a}) \\ &= e(\Pi_{\iota\in I}H(FID_\iota)^{v_\iota}\cdot g^{\sum_{\iota\in I}v_\iota m_\iota}, upk_a) \\ &= e(\Pi_{\iota\in I}H(FID_\iota)^{v_\iota}\cdot g^{\mu}, upk_a). \end{aligned} \tag{4}$$

Therefore, the integrity of the stored file is guaranteed when the cloud server indeed stores the correct file signature and follows the protocol. □

Theorem 3 (Data Privacy): A public verifier cannot learn any knowledge of the challenged blocks during the verification of data integrity.

Proof: During the integrity checking, the public auditor sends challenge message $chall = (c, k_1, k_2), 1 \le c \le n$ to the cloud server to obtain the proof message $Pro = \{\{FID_\iota\}_{\iota\in I}, \mu, \varphi\}$. If the public verifier gets two combined message $\mu = \sum_{i\in I} v_i m_i$ by pre-designing, the verifier can get the content of data by collecting a sufficient number of linear combinations of message signatures and solving the resultant linear equations [25]. For example, in the first checking task, the public verifier asks the cloud server to generate the proof message $p_1 = v_i m_i, i \in [1, c]$ with random number set as $\{v_1, v_2, \ldots, v_i, \ldots, v_c\}$ selected by the verifier. Then, the verifier initiates the second validation with random number set as $\{v_1, v_2, \ldots, v'_i, \ldots, v_c\}$ by chanding an element of I to challenge the same file message. Finally, the verifier gets two proof messages as $p_1 = v_1m_1 + \cdots + v_im_i + \cdots + v_cm_c$ and $p_2 = v_1m_1 + \cdots + v'_im_i + \cdots + v_cm_c$. Therefore, the verifier has the ability to compute the file message as $m_i = (p_2 - p_1)/(v'_i - v_i)$. To avoid the above problem of message leakage, we use a pseudo-random permutation and a pseudo-random function to genearte the message of challeng information to avoid the above situation. In this paper, if the public verifier wants to solve bilinear equations, he/she has to change an element of $\{v_i\}_{i\in I}$ generated by $\iota = \pi_{k_1}(j), v_\iota = f_{k_2}(j), 1 \le j \le c$. However, whether he/she changes the value of k_1 or k_2, the verifier cannot change only one element of $\{v_\iota\}_{\iota\in I}$. In other words, the attacker cannot get the m by

solving a set of linear equations. In this manner, the proposed schem protects data privacy in the integrity checking. □

Theorem 4 (Availabilty): The user is able to obtain the incentive value in the proposed scheme.

Proof: In algorithm **ASet**, the cloud server forms attribute chain values of user u_a as $A_a = a_1||a_2||\cdots||a_n$ based on the the availability incentive set of user u_a. Then the cloud server calculates $C_1 = g^r \in G_1$ and $C_2 = upk_a^r \cdot A_a$ and sends the encrypted availability incentive (C_1, C_2) to user u_a. With the encrypted availability incentive (C_1, C_2), user u_a is able to use his/her secrect key to decrypt it to get the value of the attribute set A_a. With the user key pair (usk_a, upk_a) of user u_a, based on the property of ElGamal encryption in Section III we have the equation as follows.

$$C_2 \cdot (C_1^{usk_a})^{-1} = \frac{C_2}{C_1^{usk_a}} = \frac{upk_a^r \cdot A_a}{g^{r \cdot usk_a}} = A_a$$

Therefore, user u_a is allowed to get the attribute set to further access the file. □

VII. PERFORMANCE EVALUATION

In this section, we first introduce some knowledge of performance evaluation. Then, we analyze the communication and computation costs of the proposed scheme, and also evaluate the performance in experiments.

A. PRIOR KNOWLEDGE

On the deployment of experiments, we adopted the Pairing Based Cryptography (PBC) [26] library to simulate the cryptographic operations in our proposed scheme. All the experiments are tested on an Intel Core i7 processor of 3.40 GHz, and the Ubuntu operating system. Without loss of generality, we test our scheme over 1,000 times.

Before detailing the performance evaluation, it's necessary to find out the relationship between the number of challenging blocks and the probability of corruption detection. In each verification task, the public verifier challenges c blocks and the cloud server responds the challenge message by generating a proof message of data possession. The value of c is very important because it determines the computaion overhead, communication overhead and the success rate of verification.

Suppose that a file F contains n blocks, where l blocks have been corrupted. The public verifier challenges c blocks to check the integrity of file F. The probability of finding one or more corrupted blocks is P_X, where X denotes the number of invalid blocks being challenged. Then we have

$$\begin{aligned} P_X &= P\{X \geq 1\} = 1 - P\{X = 0\} \\ &= 1 - \frac{n-l}{n} \times \frac{n-l-1}{n-1} \times \cdots \frac{n-l-c+1}{n-c+1} \end{aligned} \tag{5}$$

With $\frac{n-l-i}{n-i} > \frac{n-l-i-1}{n-i-1}$ and (5), we have

$$1 - (\frac{n-l}{n})^c \leq P_X \leq (\frac{n-l-c+1}{n-c+1})^c.$$

In this paper, we set the number of file blocks n as 1000 and we also show the relationship between l, the number of corrupted blocks and P_X, the probability of corruption detection when the number of challenged blocks c is 40 and 50 respectively in Fig. 4. The horizontal coordinates represent the number of corrupted blocks l and the ordinate represents the probability of detecting file corruption P_X. For example, in the orange line, the probability of corruption detection is 98.4% when we challenge 50 blocks ($c = 50$) and the number of corrupted file blocks is $l = 80$.

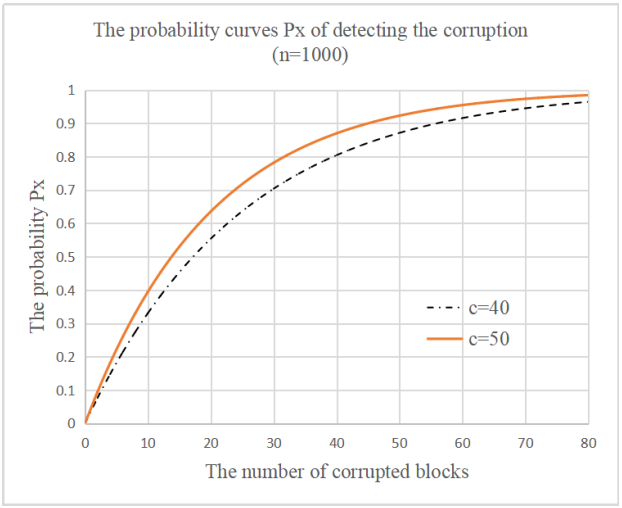

FIGURE 4. The probability curves Px of detecting the corruption.

B. THE COMMUNICATION OVERHEAD

Based on the generation of the challenge-and-response interaction in public auditing, we give the communication cost of the public verification. Before detailing the simulation, we introduce the meanings of the notations: t is the threshold value, n is the number of blocks in file F, c is the number of challenging blocks, and $|\ |$ means the length.

During the proof generation in public verification, the verifier generates a challenge message once getting the verify request from a user. According to the received challenge message $chal = (c, k_1, k_2)$, the cloud server first generates the proof message $Pro = \{\{FID_\iota\}_{\iota \in I}, \mu, \varphi\}$. The communication overhead for transmitting the challenge message $chal = (c, k_1, k_2)$, $1 \leq c \leq n$ is $3|Z_q|$. After getting the challenge message, the public verifier generates the proof message Pro which needs the communication overhead of $c|Z_q| + 2|G_1|$. The computation overhead of public auditing is summarized in Table 2.

TABLE 2. The Comparison of Communication Overhead.

	Challenge	Proof	Total
Our	$3\|Z_q\|$	$c\|Z_q\| + 2\|G_1\|$	$(c+3)\|Z_q\| + 2\|G_1\|$
CDS [14]	$(c+1)\|Z_q\|$	$c\|Z_q\| + 2\|G_1\|$	$(2c+1)\|Z_q\| + 2\|G_1\|$

CDS [14] utilizes the knowledge of blockchain and the access structure based on weighted attributes to get a convincing customized data sharing scheme. CDS also ensures the

TABLE 3. The Comparison of Computation Overhead.

	Encryption	Decryption	Proof Generation	Proof Verification	Signature
Our	$Mul_{G_1} + 2Exp_{G_1}$	$Mul_{G_1} + Exp_{G_1}$	$c\pi + cf + cMul_{Z_q}$ $+(c-1)Mul_{G_1}$ $+cExp_{G_1}$	$c\pi + cf + cMul_{G_1} +$ $2Pair + cHash$ $+(c+1)Exp_{G_1}$	$Hash + Mul_{G_1}$ $+2Exp_{G_1}$
CDS [14]	$nEn + Mul_{G_1}$	$(2t_{j,\varepsilon} - 1)$ $Mul_{Z_p} + XoR$ $+F_2$	$cMul_{Z_q}$ $+2cMul_{G_1}$	$2cMul_{Z_q} + cHash$ $+2Pair + Mul_{G_1}$ $+2Exp_{G_1}$	$Mul_{Z_p} + Mul_{G_1} +$ $Hash + 2F_1 + F_2 +$ $XoR + En$

data integrity by public auditing. In the data access method of CDS, the threshold of access is decided by the owners to realize customization. However, the communication overhead is a little high and the access method is complex, thus CDS is not suitable for the scenario proposed in this paper.

In the comparison scheme CDS [14], the public verifier has to select a c-element set to determine the challenged blocks. As shown in Table 2, CDS needs the communication overhead of $(c+1)|Z_q|$ in the transmition of challenge message. In our proposed scheme, we choose two random numbers $k_1, k_2 \in Z_q$, use a pseudo-random permutation π and a pseudo-random function f to generate identify the challenged blocks. Therefore, we need less communication overhead than CDS.

C. THE COMPUTATION OVERHEAD

In the computation overhead analysis, we use Exp_{G_1} to denote the computation time of exponentiation operation in G_1, Mul_{Z_q} and Mul_{G_1} to denote the computation time of multiplication operation in Z_p and G_1, respectively, *Hash* to denote the computation cost of a Hash operation in G_1, *En* to denote the computation time of an encryption, *XOR* to denote the computation time of a xor operation, F_1 to denote the computation time of a pseudo-random permutation, F_2 to denote the computation time of a pseudo-random function, and *Pair* to denote the computation cost of a pair operation.

As shown in Table 3, we analyze the computation overhead from the following five aspects: signature generation, proof generation, proof verification, encryption of access key and decryption of access key. In **Sign**, the user calculates the signature as $\sigma_i = (H(FID_i) \cdot g^{m_i})^{usk_a} \in G_1$ with the computation overhead of $Hash + Mul_{G_1} + 2Exp_{G_1}$. The generation of proof message needs the cost of $c\pi + cf + cMul_{Z_q} + (c-1)Mul_{G_1} + cExp_{G_1}$ to generate the proof message $Pro = \{\{FID_\iota\}_{\iota \in I}, \mu, \varphi\}$ in **ProofGen**. In **ProofVer**, the public verifier computes $\iota = \pi_{k_1}(j), v_\iota = f_{k_2}(j), 1 \le j \le c$ with the cost of $c\pi + cf$ and then checks the proof by $e(\varphi, g) = e(\Pi_{\iota \in I} H(FID_\iota)^{v_\iota} \cdot g^{\mu}, upk_a)$ with the cost of $cMul_{G_1} + 2Pair + cHash + (c+1)Exp_{G_1}$. The computation overhead of access key encryption in **ASet** and decryption in **GetA** are $Mul_{G_1} + 2Exp_{G_1}$ and $Mul_{G_1} + Exp_{G_1}$ respectively.

In addition to theoretical analysis in Table 3, we also evaluate the performance from the above five parts as shown in Fig. 5. In order to facilitate the reading, we display the overlapped places in Fig. 5. It's easy to find that the cost in proof generation and proof verification is linear with the number of

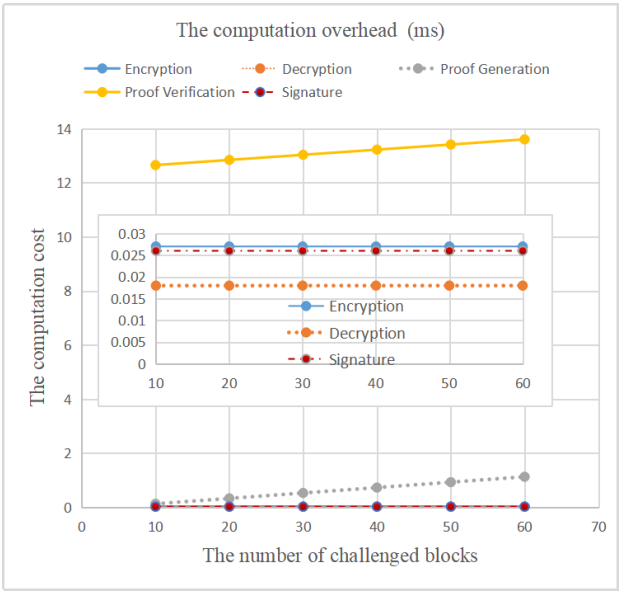

FIGURE 5. The computation cost.

the challenged blocks and that of signature, encryption and decryption is independent of the challenged number.

In the evaluation comparison of computation overhead, we compare the computation overhead of our scheme and CDS in the verification and access as the difference in the generation of signature is not significant. The comparison results of verification in Fig. 6 clearly show that the computation overhead of our scheme is a little more than that of CDS. The reason is that we use a pseudo-random permutation and a pseudo-random function to genearte the message of

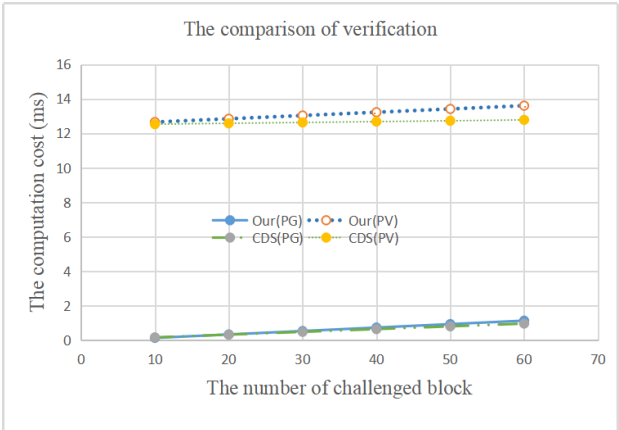

FIGURE 6. The comparison of verification.

challeng information. The advantages are high security and low communication overhead. Therefore, the a little more overhead is tolerable.

What's more, we reduce the computaion overhead of access in this paper with the help of ElGamal encryption. The comparison results (see Fig. 7) clearly demonstrate that the computation overhead of our scheme is lower than that of CDS as it uses a mathematical method by a polynomial $p(z) = a_0 + a_1 z + \cdots + a_{\tau-1} z^{\tau-1}$ which makes the cost be linear with the value of the threshold t. Therefore, our scheme is more efficient in access and meets the requirement of CPSS.

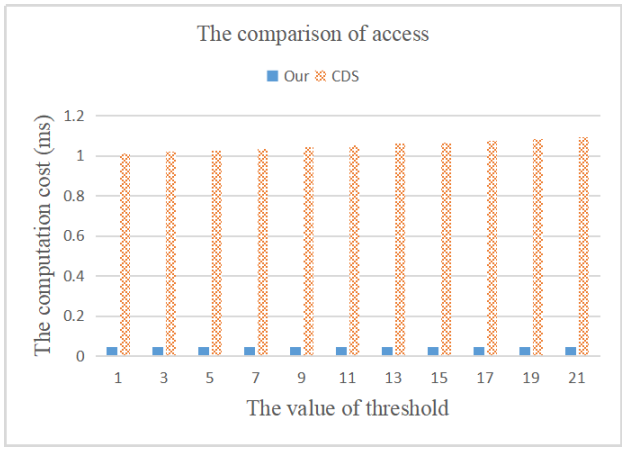

FIGURE 7. The comparison of access.

VIII. SUMMARY AND FUTURE WORK

In this paper, we propose a lightweight and secure data storage and sharing scheme that supports real-time data collection, data storage, public auditing and data access, especially for CPSS. The main contribution of the proposed scheme is our development of an incentive based data collection by giving users the weight of accessing the user, which encourages the user to take part in data collection and realizes the requirements such as real-time and reliability of CPSS. During the integrity verification of files, the proposed scheme adopts the pseudo-random method to protect data privacy against the honest-but-curious verifier. In order to realize secure and lightweight access in CPSS, we introduce ElGamal encryption to transmit the sccess key value. Extensive theoretical analyses and experimental results show the effectiveness of our proposed scheme.

This paper focuses on secure and efficient data storage and access in CPSS. In order to adapt to the requirements of CPSS, in the future work, we will focus on practical data access control, e.g., dynamic visitor groups and fair incentive mechanism. We mainly consider drawing on incentive mechanisms in the areas of blockchain and crowd sensing, and improving their dynamic performance.

REFERENCES

[1] X. Wang, L. T. Yang, X. Xie, J. Jin, and M. J. Deen, "A cloud-edge computing framework for cyber-physical-social services," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 80–85, Nov. 2017.

[2] R. Dautov, S. Distefano, D. Bruneo, F. Longo, G. Merlino, and A. Puliafito, "Data processing in cyber-physical-social systems through edge computing," *IEEE Access*, vol. 6, pp. 29822–29835, 2018.

[3] L. T. Yang, X. Wang, X. Chen, L. Wang, R. Ranjan, X. Chen, and M. J. Deen, "A multi-order distributed HOSVD with its incremental computing for big services in cyber-physical-social systems," *IEEE Trans. Big Data*, to be published.

[4] X. Wang, L. T. Yang, Y. Wang, X. Liu, Q. Zhang, and M. J. Deen, "A distributed tensor-train decomposition method for cyber-physical-social services," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 4, pp. 1–15, Oct. 2019.

[5] S. Yu, "Big privacy: Challenges and opportunities of privacy study in the age of big data," *IEEE Access*, vol. 4, pp. 2751–2763, 2016.

[6] S. Yu, G. Wang, X. Liu, and J. Niu, "Security and privacy in the age of the smart Internet of Things: An overview from a networking perspective," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 14–18, Sep. 2018.

[7] 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.

[8] Z. Su, Q. Qi, Q. Xu, S. Guo, and X. Wang, "Incentive scheme for cyber physical social systems based on user behaviors," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[9] K. Ota, M. Dong, J. Gui, and A. Liu, "QUOIN: Incentive mechanisms for crowd sensing networks," *IEEE Netw.*, vol. 32, no. 2, pp. 114–119, Mar. 2018.

[10] J. Zhou, X. S. Hu, Y. Ma, J. Sun, T. Wei, and S. Hu, "Improving availability of multicore real-time systems suffering both permanent and transient faults," *IEEE Trans. Comput.*, vol. 68, no. 12, pp. 1785–1801, Dec. 2019.

[11] J. Zhou, J. Yan, K. Cao, Y. Tan, T. Wei, M. Chen, G. Zhang, X. Chen, and S. Hu, "Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous MPSoCs," *J. Syst. Archit.*, vol. 82, pp. 1–11, Jan. 2018.

[12] J. Zhou, J. Sun, X. Zhou, T. Wei, M. Chen, S. Hu, and X. S. Hu, "Resource management for improving soft-error and lifetime reliability of real-time MPSoCs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2215–2228, Dec. 2019.

[13] S. Wang, K. Liang, J. K. Liu, J. Chen, J. Yu, and W. Xie, "Attribute-based data sharing scheme revisited in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 8, pp. 1661–1673, Aug. 2016.

[14] L. Huang, G. Zhang, S. Yu, A. Fu, and J. Yearwood, "Customized data sharing scheme based on blockchain and weighted attribute," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 206–212.

[15] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[16] 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.

[17] Y. Zhang, X.-Y. Li, and Z. Han, "Third party auditing for service assurance in cloud computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[18] Y. Yu, M. H. Au, Y. Mu, S. Tang, J. Ren, W. Susilo, and L. Dong, "Enhanced privacy of a remote data integrity-checking protocol for secure cloud storage," *Int. J. Inf. Secur.*, vol. 14, no. 4, pp. 307–318, Aug. 2015.

[19] H. Wang, D. He, J. Yu, and Z. Wang, "Incentive and unconditionally anonymous identity-based public provable data possession," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 824–835, Sep. 2019.

[20] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2271–2282, Oct. 2013.

[21] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Springer, 2005, pp. 457–473.

[22] Y. Tsiounis and M. Yung, "On the security of Elgamal based encryption," in *Proc. Int. Workshop Public Key Cryptogr.* Springer, 1998, pp. 117–134.

[23] B. Dan, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 514–532.

[24] B. Wang, B. Li, and H. Li, "Knox: Privacy-preserving auditing for shared data with large groups in the cloud," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.*, 2012, pp. 507–525.

[25] 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. 525–533.

[26] B. Lynn. (2013). *The Pairing-Based Cryptography Library*. [Online]. Available: http://crypto.stanford.edu/pbc

LONGXIA HUANG received the Ph.D. degree from the Nanjing University of Science and Technology, in 2019. She is currently a Lecturer with the School of Computer Science and Communication Engineering, Jiangsu University. Her current research interests include blockchain technology, privacy protection, data sharing, and cloud computing.

GONGXUAN ZHANG (Senior Member, IEEE) received the B.S. degree in electronic computer from Tianjin University, in 1983, and the M.S. and Ph.D. degrees in computer application from the Nanjing University of Science and Technology, in 1991 and 2005, respectively. He was a Senior Visiting Scholar with the Royal Melbourne Institute of Technology, from 2001 to 2002, and with the University of Notre Dame, from 2017 to 2017. Since 1991, he has been with the Nanjing University of Science and Technology, where he is currently a Professor with the School of Computer Science and Engineering. His current research interests include multicore and parallel processing, distributed computing, and cyber space security. He has served on the program committees in a couple of conferences. He has over 80 publications and has led or participated in 40 research projects supported by the National Science Foundation of China and the Provincial Science Foundation of Jiangsu.

SHUI YU (Senior Member, IEEE) is currently a Professor with the School of Computer Science, University of Technology Sydney, Australia. He has published two monographs and edited two books, more than 300 technical papers, including top journals and top conferences, such as IEEE TPDS, TC, TIFS, TMC, TKDE, TETC, ToN, and INFOCOM. He initiated the research field of networking for big data, in 2013. His H-index is 42. His research interests include big data, security and privacy, networking, and mathematical modelling. He is a member of AAAS and ACM. He is currently serving a number of prestigious editorial boards, including the IEEE Communications Surveys and Tutorials as an Area Editor and the *IEEE Communications Magazine*. He is a Distinguished Lecturer of IEEE Communication Society.