PAPER *Special Section on Blockchain Systems and Applications*

A Privacy-Preserving Mobile Crowdsensing Scheme Based on Blockchain and Trusted Execution Environment

Tao PENG[†a)], ***Member***, **Kejian GUAN**[†], ***and*** **Jierong LIU**[†], ***Nonmembers***

SUMMARY A mobile crowdsensing system (MCS) utilizes a crowd of users to collect large-scale data using their mobile devices efficiently. The collected data are usually linked with sensitive information, raising the concerns of user privacy leakage. To date, many approaches have been proposed to protect the users' privacy, with the majority relying on a centralized structure, which poses though attack and intrusion vulnerability. Some studies build a distributed platform exploiting a blockchain-type solution, which still requires a fully trusted third party (TTP) to manage a reliable reward distribution in the MCS. Spurred by the deficiencies of current methods, we propose a distributed user privacy protection structure that combines blockchain and a trusted execution environment (TEE). The proposed architecture successfully manages the users' privacy protection and an accurate reward distribution without requiring a TTP. This is because the encryption algorithms ensure data confidentiality and uncouple the correlation between the users' identity and the sensitive information in the collected data. Accordingly, the smart contract signature is used to manage the user deposit and verify the data. Extensive comparative experiments verify the efficiency and effectiveness of the proposed combined blockchain and TEE scheme.

key words: *mobile crowdsensing system, blockchain, trusted execution environment, privacy preservation, encryption*

1. Introduction

With the popularization and development of mobile devices, MCS [1], [2] have received significant attention, as such systems provide a platform to collect various kinds of data efficiently. The collected data usually contribute to human-centric service delivery or public social services, such as traffic congestion detection, noise detection, and water flow quality detection. MCS builds an interactive and participatory network without requiring installing any special sensors. The task publisher (TP) in an MCS can publish tasks to recruit users to collect the required data using mobile terminals. For example, the TP can collect data, analyze their information content and share knowledge with professionals or government departments through the MCS applications. A major advantage of MCS is significantly reducing the cost of traditional data collection. However, the success of MCS largely depends on the participating users. Thus, some incentive mechanisms [3] have been embedded in MCS to attract more participants with higher enthusiasm, e.g., entertainment, services, and money (usually virtual currency). Users get rewards from the TP given they provide sensitive information, such as a bank card number, shipping address, or their real name, which affords TP or other malicious attackers tracking users easily [4]–[6]. Nevertheless, information leakage exposes the users' privacy, making them less active in participating in MCS [7].

Several approaches have been proposed to protect the users' privacy [8], [9], i.e., differential privacy [10], [11], edge of computing [12]–[14], cryptographic methods [15], [16] and federated learning [8]. Most researches rely on a centralized structure, which is vulnerable to attacks and intrusions [17]. Other studies introduce a blockchain to build a distributed platform that requires a fully trusted third party (TTP) to reach a reliable reward distribution in the MCS. For example, Yang *et al.* [18] propose a blockchain privacy protection MCS by introducing a private blockchain to distribute worker transaction records and prevent attacks through reidentification. Zou *et al.* [19] suggest a blockchain-based crowd-sensing model, entitled Crowd-BLPS, for user location privacy protection. Nevertheless, most current solutions partially address the privacy concerns in an MCS and are unable to simultaneously consider the privacy preservation requirement throughout all the MCS stages, from data upload to reward distribution.

Spurred by the current research gap, this paper will develop a distributed privacy protection scheme, which is appropriate for an MCS that combines blockchain technology and a trusted execution environment (TEE) to secure privacy during data upload and reward distribution. Specifically, we employ the immutable blockchain modification to facilitate the data storage and verification and utilize the smart contract to manage the deposit and verify the data. Accordingly, TEE neglects the TTP dependence and ensures our architecture's security. Symmetric and asymmetric encryption technology [20] is exploited to process data and rewards and prevent attackers from tracking the users through the uploaded data. Our main contributions are as follows:

- Employing the TEE technology to build an isolated and trusted center and extend the trust feature to the entire network.
- Utilizing the key signature technology to verify the data integrity and exploit blockchain technology to achieve its unified consensus.
- Exploiting encryption algorithms for sensory data can achieve privacy protection for the entire data life-cycle and attain a precise reward distribution.

It should be noted that despite several types of MCS

Manuscript received April 2, 2021.
Manuscript revised June 25, 2021.
Manuscript publicized September 15, 2021.

[†]The authors are with the School of Computer Sciences and Cyber Engineering, Guangzhou University, Guangzhou, China.
a) E-mail: pengtao_work@163.com

DOI: 10.1587/transinf.2021BCP0001

attacks exist, in this work, we focus on the correlation and forgery attacks and highlight the effectiveness of our system to defend against these attacks.

The rest of this paper is structured as follows: In Sect. 2, we introduce the basic knowledge of *blockchain* and TEE. We describe the architecture of our entire system in the Sect. 3. In Sect. 4, we analyzed several attacks against our system, we conduct experiments in the Sect. 5. Related work discussions are in the Sect. 6, and the conclusions are in Sect. 7.

2. Preliminaries

2.1 Blockchain

Blockchain is a decentralized distributed network proposed by Nakamoto in 2008 for bitcoin transactions [21]. It integrates cryptography, consensus algorithm, P2P (peer-to-peer), linked list structure, and several other technologies. Blockchain [22], [23] has the characteristics of immutability, traceability, and decentralization and can be employed to solve the trust problem of digital currency under a distributed structure. In the blockchain, each user, as a network node, holds the transaction information of the entire network, which will be stored in a block in the form of a linked sequence list. Each block comprises a block body and a block header. The block body content is a Merkle tree composed of the hash value of the transaction information, and the block header includes a signature, block generation timestamp, and other information. Additionally to the first block, the block header of each block also contains the hash value of the previous block, and all blocks are formed by the direction of the hash value of the previous block linked list structure. The newly generated transaction information will be stored in the new block, and nodes randomly selected by the consensus algorithm create and merge the created blocks into the original blockchain.

2.2 TEE

A trusted execution environment (TEE) is a secure, integrity-protected execution environment consisting of memory and storage functions, which can ensure that its computing is not interfered with conventional operating systems [24]. It is commonly used to calculate and manage some sensitive data, such as fingerprint verification, mobile payment, storage of private keys and certificates, and co-exists on a device with the Rich OS, i.e., the rich text operating system, such as Android. While providing security services to Rich OS, TEE has its own execution space and affords a higher level of security than Rich OS [25], [26]. The software and hardware resources accessible by the TEE are isolated from the Rich OS, providing a secure execution environment protecting confidentiality, integrity, and access rights on the resources it accesses. The TEE mainly provides Application Programming Interfaces (APIs) such as key management, cryptographic algorithms, secure storage, and services.

3. Privacy-Preserving Mobile Crowdsensing Scheme Based on Blockchain and Trusted Execution Environment

3.1 Overview

In a traditional MCS, users upload sensory data directly to the TP. However, sensitive information, e.g., geographic location and pictures, implicit in the sensory data may expose the user's privacy to the TP or the attacker, allowing both these parties to track specific users. Additionally, based on the quality of sensory data, TP distributes incentives, e.g., money, and services, to third parties [8], while the user is rewarded by the TP each time he provides further sensitive information, such as real name or bank card details. The summary of notations used in the our scheme is shown in Table 1.

To contribute towards solving the privacy leakage problems in an MCS, our method focuses on the privacy leakage risks during data uploading and reward distribution and proposes a user privacy protection scheme based on TEE and blockchain (Fig. 1). We use TEE to build an isolated and trusted central environment and extend the credibility of TEE to the entire system by deploying multiple smart contracts in the blockchain to verify hash values and signatures. Additionally, we use encryption technology to process data and rewards, preventing attackers from tracking relevant users. Our scheme mainly involves four parties: the user, TP, TEE, and the blockchain.

- ***User***: The user is responsible for collecting the sensory data required by the TP and pre-processing them. During the data upload stage, the user first encrypts the sensory data with his user key, Key_u, and then computes the hash value of the encrypted data, $H(CT_u)$. He

Table 1 Summary of notations

Notation	*Description*
Key_u	The key customized by the user in the description file.
$H(\cdot)$	Collision-resistant hash function.
RN_u	The reward name customized by the user in the description file.
CT_u	The result of AES256 encryption using Key_u on the preprocessed data.
CT_r	The result of AES256 encryption of the reward using Key_u.
Pk_{SC}/Sk_{SC}	Public and Private Key for Smart Contracts.
Pk_{TEE}/Sk_{TEE}	Public and Private Key for TEE.

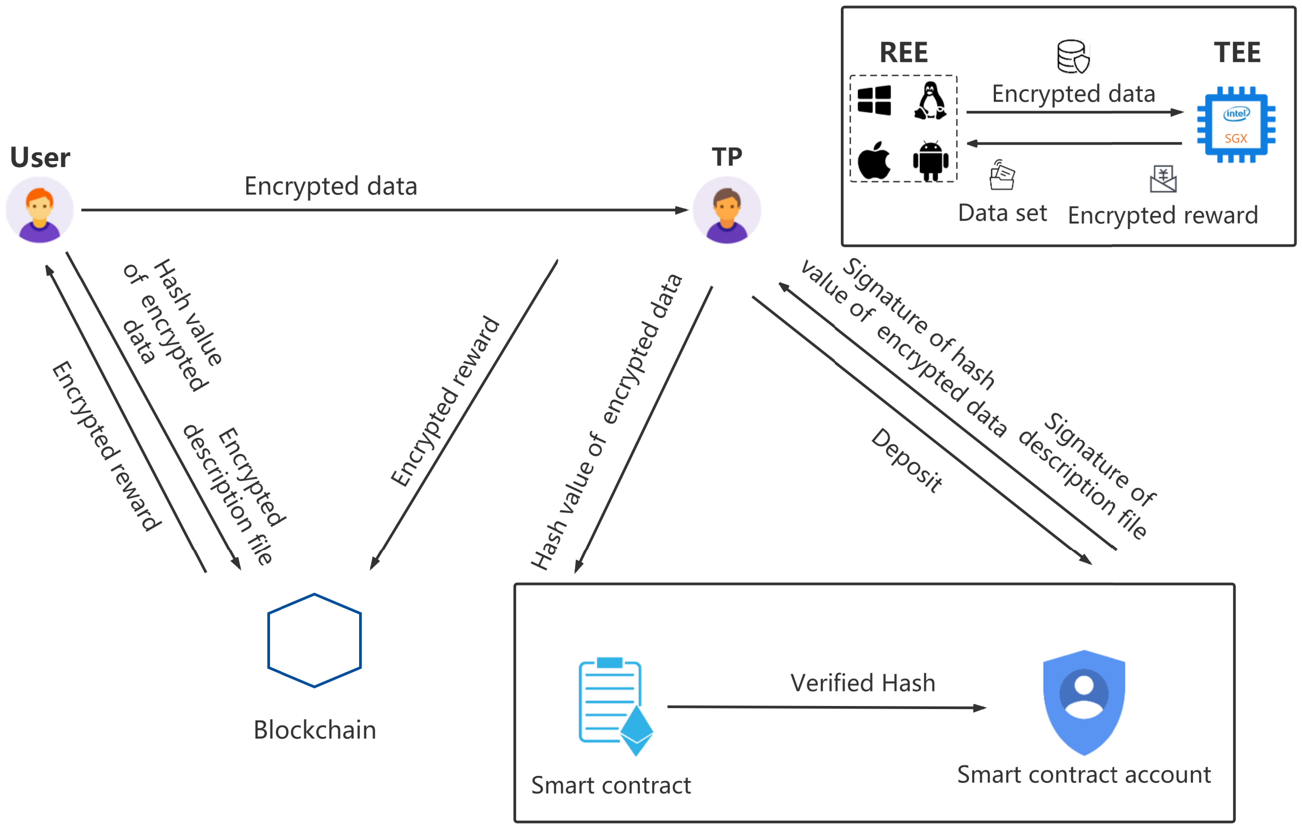

Fig. 1 The framework of our proposed scheme

also defines a description file (please refer to Sect. 3.2 for details), which is exploited to decrypt and access his rewards from the blockchain.

- ***TP***: TP's objective is to obtain from the users their sensory data with high availability, provide the user some reward, and further exploit the sensory data.
- ***TEE***: This is an isolated and trusted environment deployed on the TP that is used to calculate and manage some sensitive data to protect the user's privacy.
- ***Blockchain***: Blockchain provides reliable verification operations. During the data uploading stage, the blockchain is responsible for storing the hash value of the encrypted data along with the encrypted description file that the user uploads. Finally, the smart contract deployed in the blockchain is responsible for verifying the hash value. Considering the reward distribution process, the blockchain is responsible for storing the encrypted data uploaded by TP and exploit the smart contract to verify the data originality.

3.2 Privacy-Preserving Mobile Crowdsensing Scheme

The proposed privacy protection architecture relying on the blockchain and TEE is illustrated in Fig. 2. The scheme is divided into two processes: data upload and reward distribution.

3.2.1 Data Upload

The data upload stage includes four steps: data pre-processing, deposit payment, smart contract signature, and signature verification.

Step 1. Pre-processing.

The user conducts sensory data collection as required from TP. Nevertheless, before the collected data are uploaded to the TP, these are first pre-processed as follows:

- To ensure data security and avoid data fraud, the system will define the size of user data S in each task. The user compresses the original data or fills it with obfuscated data in a specific position so as the data size equals S (Algorithm 1).
- The user encrypts his original data with his key Key_u and obtains the AES256 encryption output CT_u.
- To ensure data integrity, the user performs the SHA256 processing on CT_u to get the hash value $H(CT_u)$ so that the TP can verify the data by the hash function when it receives the data.
- Finally, the user should define a description file, which includes the Key_u, the obfuscated data, and the reward name RN_u. We use the public key of the TEE to encrypt the description file, as only the TEE can decrypt the ciphertext with its private key. Any other nodes in the blockchain cannot access the file.

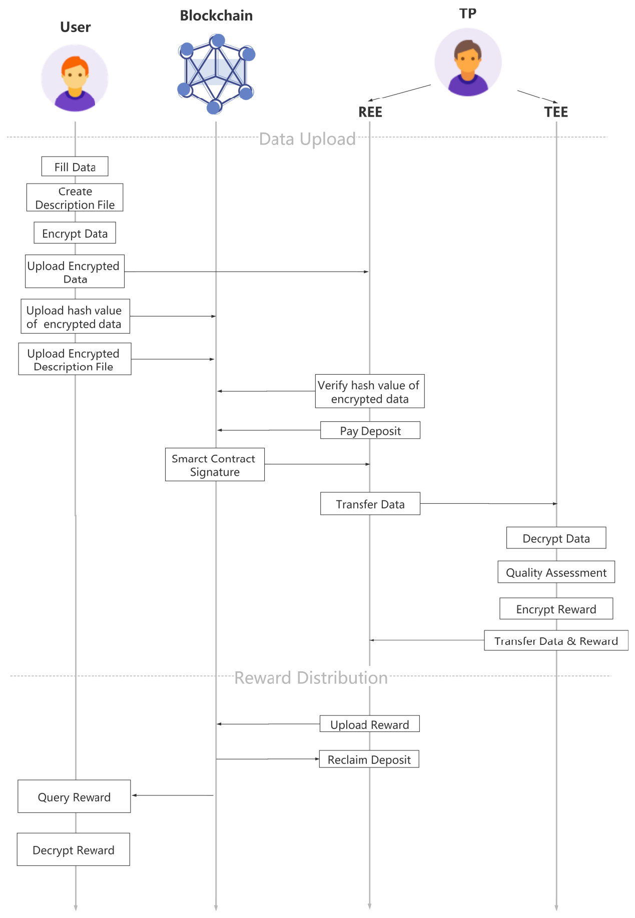

Fig. 2 The implementation process of our proposed scheme

Algorithm 1 DataSizeHandler

```
Input: The original sense data, $Data_o$; The required data size, $S$;
Output: The processed data with required size $S$, $Data_p$;
 1: $Data_o$ = NULL;
 2: $Data_t$ = NULL;
 3: $S_d = Size(Data_p)$;
 4: if $S_d == S$ then
 5:     $Data_p = Data_i$
 6: else
 7:     if $S_d > S$ then
 8:         $Data_t = CompressHandler(Data_o)$
 9:         $Data_p = DataSizeHandler(Data_t, S)$;
10:     else
11:         $S_t = S - S_d$;
12:         $Data_t = ConfuseDataGenerator(S_t)$;
13:         $Data_p = Combination(Data_p, Data_t)$;
14:     end if
15: end if
16: return $Data_p$;
```

The task type determines the size of CT_u, and in case the data size is relatively large, it is more efficient for this part of ciphertext data to be transmitted directly to TP through a centralized network. To ensure data integrity, we consider using the hash value of CT_u stored in the blockchain for the verification conducted in the TEE. Therefore, the user uploads CT_u directly to TP and uploads the $H(CT_u)$ and the encrypted description file to the blockchain.

Algorithm 2 ValidationHandler

```
Input: The public key of Smart Contract Account, $PubKey_{SC}$; The private key of Trusted Execution Environment, $PriKey_{TEE}$; The encryption of processed data, $Enc_{data}$; The hash of processed data encryption with signature, $Hash_{Enc_{data}}$; The encryption of description file with signature, $Enc_{des}$;
Output: The result of validation, $Result$;
 1: $Result = true$;
 2: $Hash_{data} = Hash(Enc_{data})$;
 3: if $!Validate(PubKey_{SC}, Hash_{Enc_{data}})$ then
 4:     $result = false$;
 5: end if
 6: if $!Validate(PubKey_{SC}, Enc_{des})$ then
 7:     $result = false$;
 8: end if
 9: if $Hash_{data}! = Hash_{Enc_{data}}$ then
10:     $result = false$;
11: end if
12: return $Result$;
```

Step 2. Deposit payment.

When TP receives CT_u from a user, it computes the hash value and compares it with the value $H(CT_u)$ uploaded by the user in the blockchain. Once they are the same, TP deposits a certain amount of virtual currency to bind TP to release rewards in the reward distribution stage. The deposits will be transferred from TP's account to the smart contract account in the blockchain and are managed by the smart contract uniformly. When TP completes the reward upload, the deposit can be refunded. While the TP does not upload a reward with a TEE signature, the smart contract will pay a certain amount of the deposit as the base reward to the user to maintain the system activity.

Step 3. Smart contract signature.

Once TP pays the deposit, the smart contract will sign $H(CT_u)$ the encrypted description file with its private key Sk_{SC}. This signature is used to prove that TP has paid the deposit and guarantees the CT_u and the encrypted description file uploaded by the user have not been tampered. Each node in the blockchain and TEE can use a public key of the smart contract Pk_{SC} to verify the authenticity. After this step is completed, TP will have CT_u, the signed $H(CT_u)$, and the encrypted description file. All these operations are performed under TP's Rich Execution Environment (REE). Given that TEE is an isolation system created by SGX and other technologies, the TP also needs to pass the CT_u uploaded by users to the TEE by calling the interface exposed by TEE.

Step 4. Signature verification.

Upon receiving the data from TP, TEE first uses the public key of the smart contract Pk_{SC} to verify $H(CT_u)$ and the encrypted description file. Then TP computes the hash value of CT_u and compares it with the received $H(CT_u)$. When either verification fails, TEE considers the data as invalid and chooses to ignore it, as shown in Algorithm 2. If the validation passes, the TEE uses Sk_{TEE} to decrypt the encrypted description file to obtain confused data, Key_u, RN_u. Now TEE can obtain the original data by decrypting the preprocessed data using Key_u.

During the data upload process, TP can track the user through specific data transfers. For example, within a certain time, TP only transmits a specific ciphertext of sensory data and waits for the plaintext of the data to return after the verification and decryption of the TEE, thereby identifying the data owner. To this end, we use K-anonymity technology to address this problem.

When the TEE obtains the original sensory data through the decryption operation, it will cache the current data, and the latter will form a freeze queue. K-anonymity protection is constructed by freezing the K data pieces in the queue so that TP cannot track the specific data user through the connection between the data ciphertext and plaintext. Then TEE performs the following judgments based on the amount of data in the current frozen queue and the task deadline:

- If the amount of data is less than K-1 and the task deadline has not been reached, TEE will freeze the sensory data obtained by the current operation and put it into the frozen data queue.
- If the amount of data reaches K-1, the sensory data obtained from the current operation is is transmitted to TP along with the data in the frozen data queue (K pieces of data in total), and the frozen data queue is cleared.
- If the amount of data is less than K-1, and the task deadline is reached, TEE generates some obfuscated data and fills the frozen data queue so that the number of frozen data queue reaches K-1, and the data obtained by the current operation along with the obfuscated data is transmitted to the TP.

In this way, the probability that TP identifies a specific user from K pieces of sensory data and establishes a correlation between the privacy information in the sensory data and the user is 1/K. Thus, as the value of parameter K increases, the probability of user privacy leakage is lower.

3.2.2 Reward Distribution

The reward distribution process includes three stages: quality evaluation, signature verification and reward collection.

Step 1. Quality evaluation.

The incentive mechanism of MCS is based on data quality and provides the corresponding rewards to the participants. In this work, we consider deploying the data quality evaluation model in TEE. Since the data quality evaluation model [27] refers to various fields such as efficiency, fairness, and privacy, the detail of implementation for the evaluation model is out of the scope of this paper. Here we assume the data submitted by all users is available, and the incentive mechanism of MCS includes various rewards such as service, entertainment, and money. This paper considers virtual currency as a reward.

Regarding the isolation feature of TEE, the virtual currency reward is deposited in a blockchain account, which will be encrypted and transmitted to the users. Each user defines the reward name RN_u and the user key Key_u in the uploaded description file. Then, TEE encrypts the blockchain address of virtual currency using AES256 with Key_u, creates the encrypted reward file CT_r, and signs it with Sk_{TEE}. Finally, TEE transmits the signed CT_r to TP, which generates key-value pairs using RN_u as the key and the signed CT_r as the value, and uploads them to the blockchain.

Algorithm 3 RewardValidationHandler

```
Input: The public key of Trusted Execution Environment, $PubKey_{TEE}$;
    The encryption of reward, $CT_r$; The custom name of reward,
    $Name_{reward}$; The account of smart contract, $SC$; The account of task
    publisher, $TP$; The base reward, $Reward_{base}$;
Output: The result of validation, $Result$;
1: $Result = true$;
2: if $!Validate(PubKey_{TEE}, CT_r)$ then
3:     $result = false$;
4: else
5:     $TP \leftarrow Reward_{base} \leftarrow SC$;
6:     $BLOCKCHAIN \leftarrow (Name_{reward} : CT_r)$;
7: end if
8: return $Result$;
```

Step 2. Signature verification.

After TP finishes the uploading process, the smart contract uses TEE's public key Pk_{TEE} to verify the reliability of CT_r, as presented in Algorithm 3. Then, the deposit will be returned to TP from the smart contract account. However, if the verification fails, the reward is considered illegitimate, and TP will not be able to reclaim the deposit.

Step 3. Reward reception.

Each reward is stored in the distributed blockchain ledger with RN_u being the key and CT_r the value. Users can query the corresponding CT_r according to RN_u and decrypt it with Key_u to get the reward.

Due to the distributed ledger feature of the blockchain, every user can query the encrypted reward files uploaded by TP, but only the owner who holds the user key can decrypt the corresponding reward. Neither the TP nor other users can detect or obtain the rewards in the ciphertext, let alone track the user's query or decrypt it. Therefore, the users' privacy and security can be guaranteed during the process of reward distribution.

4. Attack Analysis

4.1 Correlation Attack

Correlation attack by the data size. According to the volume size correlation between the data plaintext and ciphertext in the AES256 encryption algorithm, the TP can deliver to the TEE multiple data ciphertexts with significant differences in size and establish the correlation between the data ciphertext and plaintext based on the difference in size (Fig. 3). As illustrated in this example, by analyzing the data size of ciphertext and plaintext, the TP can establish the correlation between user identity, ciphertext data, and plaintext data and identify the user's identity from the sensitive information within the collected data.

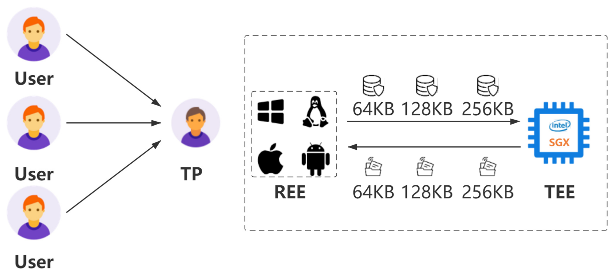

Fig. 3 The correlation attack by the data size

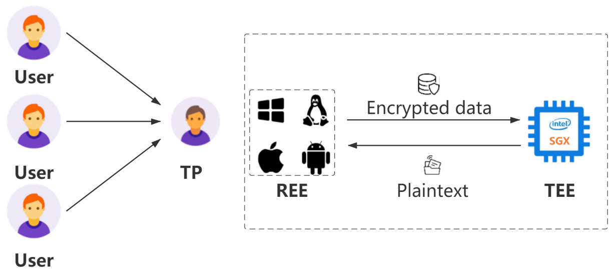

Fig. 4 Time connection between data plaintext and ciphertext

We solve this problem by using a unified data ciphertext size. In the data uploading stage, users can process the original data according to the data size required by the TP, as described in Algorithm 1. However, the size of all ciphertext data received by the TP is the same, and the transmitted plaintext data volume from the TEE to the ordinary environment cannot reveal the correlation between the ciphertext and the plaintext. Additionally, the description file that records the processing method of the sensory data size is encrypted by the RSA256 algorithm with the public key of the TEE. Therefore, the TP and other users cannot obtain the processing method of the original data from the description file and use the plaintext of the data to reverse it.

Correlation attack by transmission time. The attacker can establish the correlation among the user identity, ciphertext data, and plaintext data by analyzing the ciphertext and plaintext transmission time and identifying the user from the collected data. For example, the TP can formulate a data delivery plan, that is, within a specific time only deliver a particular piece of ciphertext and wait for the TEE to return the result, so that TP can establish a correlation between the encrypted data and the user, leading to privacy leakage, as shown in Fig. 4.

We use K-anonymity technology to solve this problem, where TP can not transmit the data to TEE until it collects K pieces of data. The TEE acts as an agent to decrypt the sensory data and deliver the original data to the TP. TEE is a system isolated from the ordinary environment, and the TP cannot control or monitor its operation. In this way, the TP can correlate the encrypted data and the user with a probability not exceeding 1/K.

4.2 Forgery Attack

The TP, as the interaction intermediary between the user and

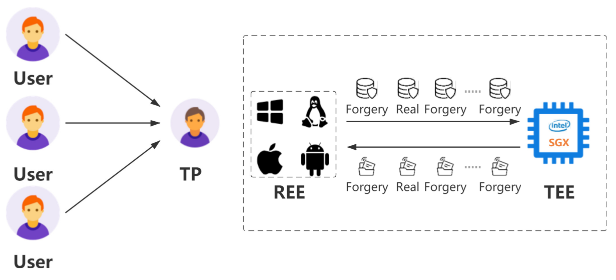
User
User
User
TP
Forgery Real Forgery ····· Forgery
Forgery Real Forgery ····· Forgery
REE
TEE
intel SGX

Fig. 5 The connection between plaintext and encrypted data is obtained by forging data

the TEE, can obtain the correlation between the specific ciphertext and the plaintext data by reducing forged data or even remove the protection of K-anonymity. As mentioned earlier, the K-anonymity method can aggregate specific data with other K-1 copies of frozen data so that the probability of TP identifying a particular user is no more than 1/K. While the aggregated data returned by the TEE contains data forged by the TP, the latter can identify the specific data. As illustrated in Fig. 5, if the K-1 copies of frozen data are all forged by the TP, the probability of TP identifying specific CT_u and data rises from 1/K to 1, and the identity of sensitive information contained in the data refers to the privacy of the corresponding user.

To address this problem, we employ private key signature technology to verify data in TEE. Upon receiving CT_u and the signed $H(CT_u)$ description file, TEE first uses the public key of smart contract Pk_{SC} to verify the signature included in the description file and checks the hash value of the ciphertext data. Therefore, TP needs to use Sk_{SC} to sign $H(CT_u)$ in the blockchain and pay the deposit to obtain the proof of signature. The data forged locally by TP will fail to verify the smart contract, and therefore it cannot be verified by the TEE signature. The verification mechanism based on private key signature technology can prevent TP from forging data, thus protecting user privacy.

5. Experiment and Result Analysis

5.1 Comparative Experiment between Common Environment and TEE

5.1.1 Experimental Environment

In the following simulations, we utilize a MacBook Pro 16 2019 that runs Ubuntu 64-bit version 20.04.1 on a virtual machine through the VMware Fusion software. The computer's hardware configuration is: Intel Core i7 2.6GHz Hexa-core, 1TB solid-state drive, 16GB 2667MHz DDR4 memory, AMD Radeon Pro 5300M 4GB graphics card, and the corresponding software setup is: macOS Big Sur version 11.1 and VMware Fusion 12.0.0. The experimental machine utilizes eight processor cores, 12GB RAM, and 50GB disk size for Ubuntu virtual machine. The experiments are conducted in the virtual machine using GCC version 9.3.0, SGX-Linux SDK, and Ippcrypto library. The experimental

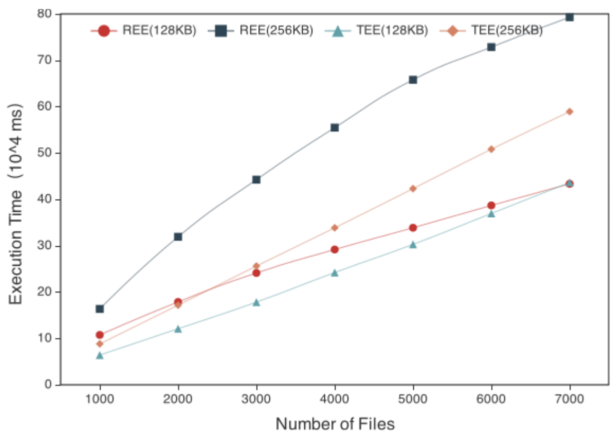

Fig. 6 Comparison of RSA256 algorithm encryption execution time between REE and TEE

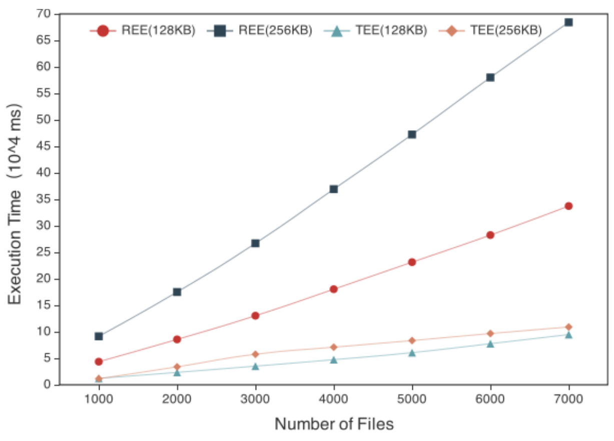

Fig. 8 Comparison of the AES256 algorithm encryption execution time between REE and TEE

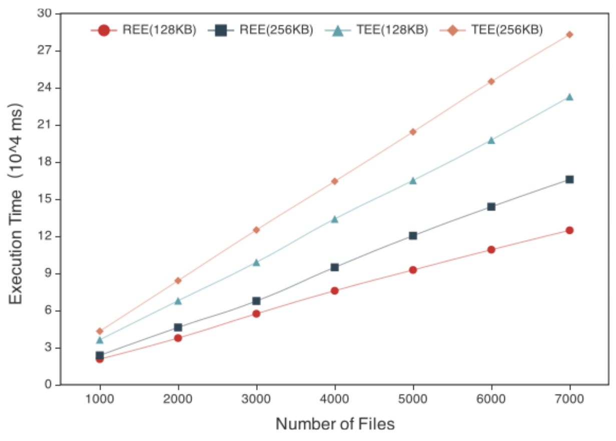

Fig. 7 Comparison of RSA256 algorithm decryption execution time between REE and TEE

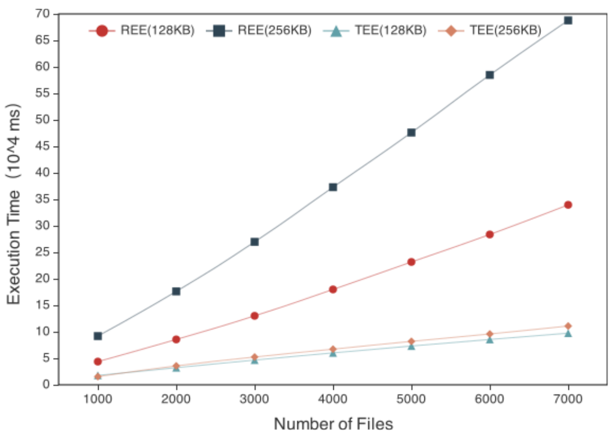

Fig. 9 Comparison of the AES256 algorithm decryption execution time between REE and TEE

data is based on 7,000 pictures from the Yelp dataset [28].

5.1.2 Experimental Result

We employ the SGX-Linux SDK to deploy a TEE in the experimental virtual machine. We compare the encryption and decryption efficiency of the RSA256 and AES256 algorithms in the normal environment and the TEE, demonstrating the impact of exploiting a TEE on the efficiency of MCS through the comparison of the two experimental data. The experimental data employs 128KB and 256KB files as the data source, and the execution time is recorded under different file numbers.

Figure 6 and Fig. 7 compare the execution time on TEE and REE using the RSA256 algorithm. Both figures highlight that as the number of processed files increases, the rising trend of RSA256 algorithm encryption execution time in REE becomes slower, and the trend in TEE remains the same. For the same number of files, the execution time of the RSA256 algorithm in REE and TEE is proportional to the file size. As a result, REE has an advantage over TEE in the execution time of encryption, while the opposite is true for the decryption time. This shows that for the RSA256 algorithm, TEE handles encryption operations more efficiently, while REE handles decryption more efficiently. On the decryption operations, the file size and number have similar effects on REE and TEE, while on encryption operations, REE performance declines faster as the number of files increases.

Figures 8 and 9 compare the execution time using the AES256 algorithm between the REE and TEE. The execution time of the AES256 algorithm in the TEE is more advantageous than the normal environment both for the encryption and decryption operations.

Figures 8 and 9 illustrate the AES256 algorithm execution time in REE and TEE, with both figures presenting similar curves. The execution time for REE and TEE increases as the number of files increases, presenting a gentle trend. The difference is that for the same number of files, the execution time of REE for 128KB and 256KB file sizes is different, while the file size has little effect on TEE. Additionally, the AES256 algorithm execution time for the TEE is superior to that of REE for both the encryption and decryption operations, and this advantage increases as the number of

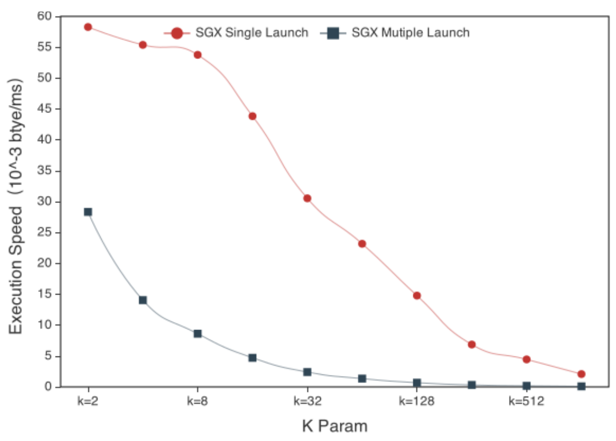

Fig. 10 Comparison of the efficiency of the SGX single start and multiple starts under different parameter K values

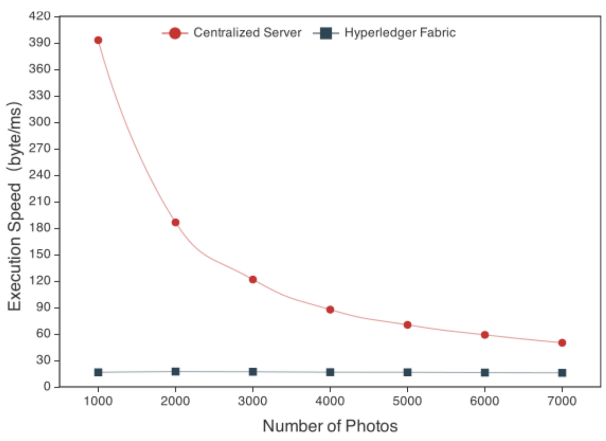

Fig. 11 Comparison of data write execution speed between a traditional centralized server and the Hyperledger Fabric alliance chain

files increase.

The results in Fig. 8 and Fig. 9 illustrate that TEE is less appealing compared to REE in the RSA256 decryption, and the gap becomes more significant as the amount of data processed by the system increases. In contrast, Fig. 8 and Fig. 9 illustrate that TEE has an advantage over REE in both AES256 algorithm encryption and decryption, and the former is less affected by data size. TEE mainly describes the file decryption, data decryption, and reward encryption operations in our proposed scheme, i.e., RSA256 decryption, AES256 decryption, and AES256 encryption. In terms of data size, the perception data is more extensive, next is the description file, and the reward is the least. Therefore, the advantage of TEE utilizing the AES256 algorithm encryption and decryption can balance its disadvantage of the RSA256 algorithm decryption to a certain extent, and it can be considered that the introduction of TEE in this paper has a negligible impact on the overall system performance.

We adopt the K-anonymity method to protect the user's privacy, with the degree of privacy protection affected by the value of K. We test the execution speed of TEE with different K values (2 to 1024) using 587byte as the data source and consider the efficiency of simultaneous transmission of K pieces of data (SGX single start) and independent transmission (SGX multiple start). As presented in Fig. 10, the SGX single-start and multiple-start execution speeds both show a decreasing trend as the K value rises, with the former having a downward trend with multiple fluctuations, and the latter being gradually slow. Additionally, SGX single-start execution speed is more advantageous than SGX multiple-start for the same K value, but the gap between the two decreases as the K fetch value rises.

5.2 Comparative Experiment of Traditional Centralized Server and Hyperledger Fabric Alliance Chain

5.2.1 Experimental Environment

To test the performance gap between traditional centralized servers and blockchains, we also use Docker technology to build a Hyperledger Fabric alliance chain on an experimental machine configured with an Intel i7-7700 (Core 3.6GHZ) CPU, 1TB 7200r/min SATA hard drive, 8GB DDR4 2400 RAM, using Ubuntu 16.04.7 system and Hyperledger Fabric official mirror. The experimental data is obtained with a certain number of pictures from the Yelp dataset [28] as the data source, and the execution speed of data transmission to the Hyperledger Fabric Alliance chain (data writing) and data acquisition from it (data reading) is recorded under different numbers of pictures.

5.2.2 Experimental Result

Figure 11 highlights that as the number of images increases, the data write execution speed of the traditional centralized server tends to decrease significantly. In contrast, the Hyperledger Fabric alliance chain remains stable. Additionally, for the same number of images, the execution speed of a traditional centralized server data writing is always faster than that of the Hyperledger Fabric alliance chain, but the gap between the two is narrowing as the number of images rises.

Figure 12 illustrates that as the number of images rises, the data read execution speed of the traditional centralized server significantly decreases. In contrast, the Hyperledger Fabric federated chain maintains a slight upward trend. Additionally, when the number of images is less than 1500, the traditional centralized server data read execution speed has an advantage over the Hyperledger Fabric. However, when the number of images exceeds 1500, the opposite is true, and the gap between the two increases as the number of images rises.

Based on the blockchain's distributed network structure, we also challenge the performance of the Hyperledger Fabric alliance chain under different numbers of nodes. As illustrated in Fig. 13, as the number of images increases, the changing trend of the Hyperledger Fabric alliance chain 4-node and 8-node networks is similar. Moreover, the former executes faster than the latter with the same number of im-

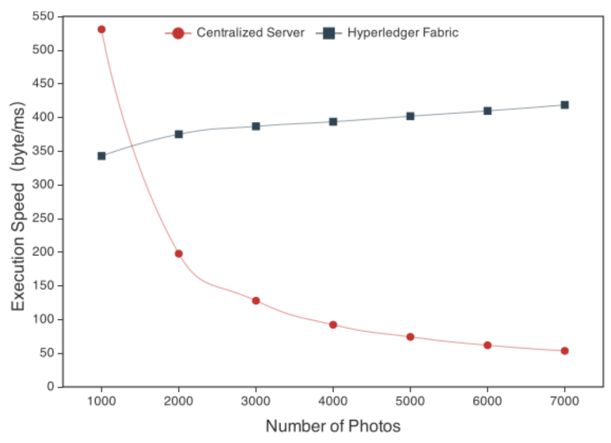

Fig. 12 Comparison of data read execution speed between a traditional centralized server and the Hyperledger Fabric alliance chain

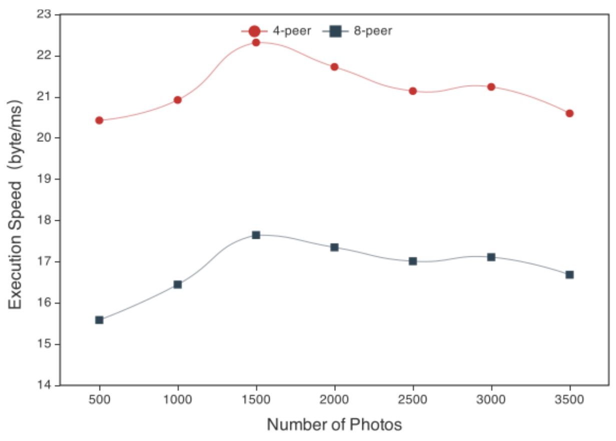

Fig. 13 Hyperledger Fabric Consortium, Chain 4-node and 8-node performance comparison

ages with a difference of about 23.5%.

Figures 11 and 12 illustrate the respective advantages of traditional centralized servers and Hyperledger Fabric federated chains in data writing and reading operations. The difference is that the performance of the former will decrease due to increasing the number of pictures, while the latter tends to be stable. For example, Fig. 13 shows that the performance of the Hyperledger Fabric alliance chain is affected more by the number of network nodes than the number of images.

6. Related Work

Recently several approaches, such as K-anonymity [29], [30], blockchain [31]–[34], differential privacy [10], federated learning [8] and edge of computing [35], and cryptographic methods [8], [9], [36]–[38] have been proposed to protect the users' privacy in an MCS. For example, Wang *et al.* [10], [39] propose a framework for protecting the users' location privacy during task distribution based on geolocation obfuscation, and also introduce diferential privacy techniques to protect users' privacy in an MCS. Shen *et al.* [40] consider using edge computing techniques to solve the privacy leakage problem in the task distribution process. Accordingly, Zhao *et al.* [8] propose a mechanism with privacy protection and quality awareness. Most of these studies are conducted in centralized servers, such as introducing a TTP for sensitive operations in K-anonymity and obfuscation techniques, which are almost non-existent in daily life. Hence, current studies are less usable in real-world applications.

Many research scholars have suggested blockchain [18], [41]–[43] to solve privacy problems in MCS. For example, Zou *et al.* [19] introduce a blockchain-based crowd-sensing model of location privacy protection, entitled CrowdBLPS, for handling task publishing, staff selection, quality assessment, and data upload. Huang *et al.* [44] combine blockchain with crowd-sense in an industrial system, while Hu *et al.* [45] propose a crowd-sense incentive mechanism based on blockchain. An *et al.* [46] provide a decentralized crowd-aware model with an anonymity policy based on an elliptic curve algorithm verification scheme that is employed to protect the users' identity privacy. Kadadha *et al.* [47] utilize the decentralized characteristics of the blockchain to enable multitasking and multi-user task assignment and to protect user privacy. In our previous work [48], we proposed an incentive mechanism combining multiple blockchains for protecting the users' privacy in an MCS. Due to the transparency of the blockchain, all users on the chain can access the data stored on the blockchain, while malicious users can use these contents to find specific users. Therefore, even though blockchain has powerful features, it still needs to be further considered and studied when it is directly used for MCS privacy protection.

TEE is another solution to the trust problem [49], [50] and is widely used in mobile payment and biometrics. Its security is guaranteed by building a system isolated from the standard operating environment at the software and hardware levels, and thus scholars have introduced TEE based on blockchain to accomplish sensitive operations. Specifically, Dai *et al.* [51] propose a lightweight wallet with an authentication mechanism to protect the users' private keys and wallet addresses based on Hyperledger Fabric. Maddali *et al.* [25] use TEE to reduce the redundant execution of smart contracts, while Zhang *et al.* [52] combine blockchain and TEE to build an electronic voting system with blind signatures and homomorphic encryption to protect the privacy of voters and voting results and reduce the blockchain load by processing cryptographic operations through TEE. Ekiden is a smart contract confidentiality and performance solution proposed by Cheng *et al.* [53]. This study concludes that the public and transparent features of the blockchain and the performance problems can hinder the development of smart contracts and move them to an off-chain environment by introducing TEE to complete smart contract execution and remotely verify the execution results. Although various privacy-preserving techniques have shown effectiveness in MCS research in recent years, these studies still only address the implementation of privacy protection for task distribution or data uploading process, without considering the impact of the solution on the implementation of incentives or the privacy leakage problem of the reward distribution process. Opposing to existing methods, our proposed privacy protection scheme is based on blockchain and TEE for MCS and realizes privacy preservation both in the process of data upload and reward distribution.

7. Conclusion

Aiming at the privacy leakage problems during the MCS data upload and reward distribution processes, this paper proposes a user privacy protection scheme based on blockchain and TEE to ensure the integrity of the sensory data and the reliability of the reward distribution. The characteristics of the blockchain, such as decentralization, immutability, tamper-proof, and transparency, afford a solution to the trust problem that exists in the centralized system. The isolation capabilities supported by the TEE in both hardware and software are sufficient to ensure the security of sensitive information. Combined with the encryption algorithm, our proposed scheme provides privacy protection, as the TP and other malicious attackers cannot obtain sensitive information about the users through correlating sensory data and user rewards. Our paper also assumes two attacks that may lead to privacy disclosure: correlation attack and forgery attack and analyzes the corresponding solutions based on the proposed scheme. Finally, two comparative experiments highlight that introducing blockchain and the trusted execution environment has little effect on the overall system efficiency. In our scheme, we do not consider the impact of the users' data quality on the system. Thus, future work shall involve research on the quality assessment model for MCS.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61802076, 61632009 and 61872097, in part by the National Key Research and Development Program of China (2020YFB1005804), in part by the Guangdong Provincial Natural Science Foundation under Grant 2017A030308006 and the Science and Technology Program Foundation of Guangdong Province under Grant PDJH2021B0398.

References

[1] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," IEEE Commun. Mag., vol.49, no.11, pp.32–39, 2011.
[2] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," IEEE communications surveys & tutorials, vol.21, no.3, pp.2419–2465, 2019.
[3] J. Blömer, J. Bobolz, D. Diemert, and F. Eidens, "Updatable Anonymous Credentials and Applications to Incentive Systems," Proc. 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19, pp.1671–1685, Association for Computing Machinery, 2019.
[4] T. Peng, J. Liu, G. Wang, Q. Liu, J. Chen, J. Zhu, and C. Huang, "A User-Defined Location-Sharing Scheme with Efficiency and Privacy in Mobile Social Networks," Scientific Programming, pp.1–13, 2020.
[5] Q. Liu, Y. Peng, J. Wu, T. Wang, and G. Wang, "Secure Multi-Keyword Fuzzy Searches with Enhanced Service Quality in Cloud Computing," IEEE Transactions on Network and Service Management, vol.18, no.2, pp.2046–2062, 2021.
[6] Q. Liu, P. Hou, G. Wang, T. Peng, and S. Zhang, "Intelligent route planning on large road networks with efficiency and privacy," Journal of Parallel and Distributed Computing, vol.133, pp.93–106, 2019.
[7] Z. Wang, X. Pang, J. Hu, W. Liu, Q. Wang, Y. Li, and H. Chen, "When mobile crowdsensing meets privacy," IEEE Commun. Mag., vol.57, no.9, pp.72–78, 2019.
[8] B. Zhao, S. Tang, X. Liu, and X. Zhang, "PACE: Privacy-Preserving and Quality-Aware Incentive Mechanism for Mobile Crowdsensing," IEEE Trans. Mobile Comput., vol.20, no.5, pp.1924–1939, 2021.
[9] D. Tao, T.-Y. Wu, S. Zhu, and M. Guizani, "Privacy protection-based incentive mechanism for Mobile Crowdsensing," Computer Communications, vol.156, pp.201–210, 2020.
[10] L. Wang, D. Zhang, D. Yang, B.Y. Lim, X. Han, and X. Ma, "Sparse mobile crowdsensing with differential and distortion location privacy," IEEE Trans. Inf. Forensics Security, vol.15, pp.1–15, 2020.
[11] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving privacy with probabilistic indistinguishability in weighted social networks," IEEE Trans. Parallel Distrib. Syst., vol.28, no.5, pp.1417–1429, 2017.
[12] T. Wang, Y. Lu, J. Wang, H.-N. Dai, X. Zheng, and W. Jia, "EIHDP: Edge-intelligent hierarchical dynamic pricing based on cloud-edge-client collaboration for IoT systems," IEEE Trans. Comput., vol.14, no.8, pp.1–14, 2021.
[13] T. Wang, P. Wang, S. Cai, X. Zheng, Y. Ma, W. Jia, and G. Wang, "Mobile edge-enabled trust evaluation for the Internet of Things," Information Fusion, vol.75, pp.90–100, 2021.
[14] L. Zhang, Y. Zou, W. Wang, Z. Jin, Y. Su, and H. Chen, "Resource allocation and trust computing for blockchain-enabled edge computing system," Computers & Security, vol.105, pp.1–13, 2021.
[15] J. Song, Q. Zhong, W. Wang, C. Su, Z. Tan, and Y. Liu, "FPDP: Flexible privacy-preserving data publishing scheme for smart agriculture," IEEE Sensors J., vol.21, no.16, pp.17430–17438, 2021.
[16] W. Wang, H. Huang, L. Zhang, and C. Su, "Secure and efficient mutual authentication protocol for smart grid under blockchain," Peer-to-Peer Networking and Applications, vol.14, no.5, pp.2681–2693, 2021.
[17] S. Boukoros, M. Humbert, S. Katzenbeisser, and C. Troncoso, "On (the Lack of) Location Privacy in Crowdsourcing Applications," Proc. 28th USENIX Conference on Security Symposium, SEC'19, pp.1–18, USENIX Association, 2019.
[18] M. Yang, T. Zhu, K. Liang, W. Zhou, and R.H. Deng, "A blockchain-based location privacy-preserving crowdsensing system," Future Generation Computer Systems, vol.94, pp.408–418, 2019.
[19] S. Zou, J. Xi, H. Wang, and G. Xu, "CrowdBLPS: A Blockchain-based Location Privacy-Preserving Mobile Crowdsensing System," IEEE Trans. Ind. Informat., vol.16, no.6, pp.4206–4218, 2020.
[20] S.S. Al-Riyami and K.G. Paterson, "Certificateless public key cryptography," 9th International Conference on the Theory and Application of Cryptology and Information Security, volume 2894 of Lecture Notes in Computer Science, pp.452–473, Springer, 2003.
[21] S. Nakamoto, "Re: Bitcoin P2P e-cash paper," The Cryptography Mailing List, pp.1–2, 2008.
[22] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," IEEE Transactions on Services Computing, vol.13, no.2, pp.1–2, 2019.
[23] Y. Xu, Q. Zeng, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism,"

Concurrency and Computation: Practice and Experience, vol.32, no.5, pp.1–7, 2020.

[24] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted execution environment: what it is, and what it is not," 2015 IEEE Trustcom/BigDataSE/ISPA, volume 1 of TRUSTCOM '15, pp.57–64, IEEE, 2015.

[25] L.P. Maddali, M.S.D. Thakur, R. Vigneswaran, M.A. Rajan, S. Kanchanapalli, and B. Das, "VeriBlock: A novel blockchain framework based on verifiable computing and trusted execution environment," 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), COMSNETS '12, pp.1–6, IEEE, 2020.

[26] Y. Wang, J. Li, S. Zhao, and F. Yu, "Hybridchain: A Novel Architecture for Confidentiality-Preserving and Performant Permissioned Blockchain Using Trusted Execution Environment," IEEE Access, vol.8, pp.190652–190662, 2020.

[27] F. Restuccia, N. Ghosh, S. Bhattacharjee, S.K. Das, and T. Melodia, "Quality of information in mobile crowdsensing: Survey and research challenges," ACM Transactions on Sensor Networks (TOSN), vol.13, no.4, pp.1–43, 2017.

[28] Yelp Dataset Challenge [EB/OL], "https://www.yelp.com/dataset/challenge.

[29] T. Peng, Q. Liu, G. Wang, Y. Xiang, and S. Chen, "Multidimensional privacy preservation in location-based services," Future Generation Computer Systems, vol.93, pp.312–326, 2019.

[30] T. Peng, Q. Liu, B. Hu, J. Liu, and J. Zhu, "Dynamic keyword search with hierarchical attributes in cloud computing," IEEE Access, vol.6, pp.68948–68960, 2018.

[31] L. Zhang, Z. Zhang, W. Wang, Z. Jin, Y. Su, and H. Chen, "Research on a covert communication model realized by using smart contracts in blockchain environment," IEEE Syst. J., pp.1–12, 2021.

[32] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," Ieee Access, vol.4, pp.2292–2303, 2016.

[33] G. Zyskind, O. Nathan, and A.'S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," 2015 IEEE Security and Privacy Workshops, pp.180–184, IEEE, 2015.

[34] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," 2016 IEEE symposium on security and privacy (SP), pp.839–858, IEEE, 2016.

[35] T. Wang, Y. Liu, X. Zheng, H.-N. Dai, W. Jia, and M. Xie, "Edge-Based Communication Optimization for Distributed Federated Learning," IEEE Transactions on Network Science and Engineering, vol.14, no.8, pp.1–11, 2021.

[36] Q. Liu, Y. Peng, S. Pei, J. Wu, T. Peng, and G. Wang, "Prime Inner Product Encoding for Effective Wildcard-based Multi-Keyword Fuzzy Search," IEEE Transactions on Services Computing, pp.1–13, 2020.

[37] Q. Liu, Y. Tian, J. Wu, T. Peng, and G. Wang, "Enabling verifiable and dynamic ranked search over outsourced data," IEEE Transactions on Services Computing, pp.1–14, 2019.

[38] D. Yuan, Q. Li, G. Li, Q. Wang, and K. Ren, "PriRadar: A privacy-preserving framework for spatial crowdsourcing," IEEE Transactions on Information, Forensics and Security, vol.15, pp.299–314, 2020.

[39] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," Proc. 26th International Conference on World Wide Web, pp.627–636, ACM, 2017.

[40] H. Shen, G. Bai, Y. Hu, and T. Wang, "P2TA: Privacy-Preserving Task Allocation for Edge Computing Enhanced Mobile Crowdsensing," Journal of Systems Architecture, vol.97, pp.130–141, 2019.

[41] Y. Xu, C. Zhang, Q. Zeng, G. Wang, J. Ren, and Y. Zhang, "Blockchain-Enabled Accountability Mechanism Against Information Leakage in Vertical Industry Services," IEEE Transactions on Network Science and Engineering, vol.8, no.2, pp.1202–1213, 2021.

[42] T. Wang, H. Luo, X. Zheng, and M. Xie, "Crowdsourcing mechanism for trust evaluation in cpcs based on intelligent mobile edge computing," ACM Transactions on Intelligent Systems and Technology (TIST), vol.10, no.6, pp.1–19, 2019.

[43] Y. Ma, Y. Sun, Y. Lei, N. Qin, and J. Lu, "A survey of blockchain technology on security, privacy, and trust in crowdsourcing services," World Wide Web, vol.23, no.1, pp.1–27, 2020.

[44] J. Huang, L. Kong, H.-N. Dai, W. Ding, L. Cheng, G. Chen, X. Jin, and P. Zeng, "Blockchain-Based Mobile Crowd Sensing in Industrial Systems," IEEE Trans. Ind. Informat., vol.16, no.10, pp.6553–6563, 2020.

[45] J. Hu, K. Yang, K. Wang, and K. Zhang, "A blockchain-based reward mechanism for mobile crowdsensing," IEEE Trans. Comput. Social Syst., vol.7, no.1, pp.178–191, 2020.

[46] J. An, H. Yang, X. Gui, W. Zhang, R. Gui, and J. Kang, "TCNS: Node Selection With Privacy Protection in Crowdsensing Based on Twice Consensuses of Blockchain," IEEE Transactions on Network and Service Management, vol.16, no.3, pp.1255–1267, 2019.

[47] M. Kadadha, H. Otrok, R. Mizouni, S. Singh, and A. Ouali, "SenseChain: A blockchain-based crowdsensing framework for multiple requesters and multiple workers," Future Generation Computer Systems, vol.105, pp.650–664, 2020.

[48] T. Peng, J. Liu, J. Chen, and G. Wang, "A Privacy-Preserving Crowdsensing System with Muti-Blockchain," 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp.1944–1949, IEEE, 2020.

[49] N. Asokan, "Hardware-assisted trusted execution environments: Look back, look ahead," Proc. 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19, p.1687, ACM, 2019.

[50] J.-E. Ekberg, K. Kostiainen, and N. Asokan, "Trusted execution environments on mobile devices," Proc. 2013 ACM SIGSAC conference on Computer & communications security, CCS '13, pp.1497–1498, ACM, 2013.

[51] W. Dai, Q. Wang, Z. Wang, X. Lin, D. Zou, and H. Jin, "Trustzone-based secure lightweight wallet for hyperledger fabric," Journal of Parallel and Distributed Computing, vol.149, pp.66–75, 2021.

[52] Y. Zhang, Y. Li, L. Fang, P. Chen, and X. Dong, "Privacy-protected Electronic Voting System Based on Blockchin and Trusted Execution Environment," 2019 IEEE 5th International Conference on Computer and Communications (ICCC), pp.1252–1257, IEEE, 2019.

[53] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. Johnson, A. Juels, A. Miller, and D. Song, "Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contracts," 2019 IEEE European Symposium on Security and Privacy (EuroS&P), pp.185–200, IEEE, 2019.

Tao Peng received the B.Sc. degree in computer science from Xiangtan University, China, in 2004,the M.Sc. degree in circuits and systems from Hunan Normal University, China, in 2007, and the Ph.D. degree in computer science from Central South University, China, in 2017. She is currently an associate professor in the School of Computer Sciences and Cyber Engineering, Guangzhou University, Guangzhou, China. Her research interests include network and information security issues.

Kejian Guan received a bachelor's degree in engineering from Foshan University in 2020 and is currently studying for a master's degree in computer science at the School of Computer Sciences and Cyber Engineering, Guangzhou University, Guangzhou, China.

Jierong Liu received the B.Sc. degree in educational technology from Guangzhou University, China, in 2017, where he is currently pursuing the master's degree with the School of Computer Sciences and Cyber Engineering, Guangzhou University, Guangzhou, China. His research interests include security and privacy issues in mobile social network.