Computer Communications 194 (2022) 44–54

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Efficient sharing of privacy-preserving sensing data on consortium blockchain via group key agreement

Xiaoyan Hu [a,b,c,*], Xiaoyi Song [a], Guang Cheng [a,d,e], Hua Wu [a], Jian Gong [a]

[a] School of Cyber Science and Engineering, Southeast University, Nanjing, 211189, China
[b] Purple Mountain Laboratories for Network and Communication Security, Nanjing, 211111, China
[c] Jiangsu Provincial Engineering Research Center of Security for Ubiquitous Network, Nanjing, 211189, China
[d] Research Base of International Cyberspace Governance (Southeast University), Nanjing, 211189, China
[e] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China

ARTICLE INFO

Keywords:
Consortium blockchain
Wireless sensor network
Privacy preservation
Data sharing
Group key agreement

ABSTRACT

Due to the self-organizing and distributed nature of Wireless Sensor Networks (WSN), valuable data in sensor networks faces security and privacy risks. A blockchain-based approach enables secure and convenient sharing of sensor information among different users. Compared to public and private blockchains, consortium blockchain is widely used across different industries and use cases in WSN due to its auditability and high transaction rate. However, sensing data sharing via consortium blockchain raises the privacy issue. Therefore, the data from the sensor node is encrypted by the key (named *sensorkey*) shared by the sensor node and the sink node and then sent to the blockchain network to not reveal the privacy of the sensing data. Since the infrastructure in large-scale WSN is usually owned and managed by multiple organizations, encrypted sensing data needs to be authorized by these multiple organizations for computation. Organizations requesting privacy-preserving data are referred to as data sharers. Distributing the *sensorkey* to each data sharer requires separate encryption of the key using the data sharer's public key. The sink node needs to be online when each data sharer asks for the *sensorkey*, and one encryption of the *sensorkey* for each data sharer consumes precious resources. This work proposes GSChain for efficient privacy-preserving sensing data sharing on consortium blockchain. Multiple data sharers resort to asymmetric group key agreement protocol to maintain a shared group encryption key and their respective group decryption keys, enabling efficient *sensorkey* retrieval from the consortium blockchain. The *sensorkey* is encrypted only once by the group encryption key and stored on the consortium blockchain along with the privacy-preserving sensing data. Our scheme improves the efficiency of privacy-preserving data sharing among multiple data sharers while reducing the online demand for sink nodes. Although the data-sharing group should remain stable for a long time, we design the group key update scheme. We also discuss how old and new data sharers access different ranges of privacy-preserving sensing data as the data-sharing group changes. We build a complete implementation of GSChain based on the Hyperledger Fabric framework and conduct a comprehensive set of experimental studies. Our experimental results demonstrate that GSChain improves the privacy-preserving sensing data sharing efficiency with tolerable time and storage overhead. In addition, the time overhead caused by the recovery of the GSChain system is tolerable when the membership of the data-sharing group changes.

1. Introduction

As an essential part of the Internet of Things (IoT), Wireless Sensor Network (WSN) has broad application prospects in national defense and military, smart home, and environmental monitoring. The data collected by sensors facilitates us to work smart, but there is a risk of privacy leakage due to the inclusion of sensitive information. For example, wireless sensors are deployed in field areas to monitor ambient temperature and humidity. The collected and queried data often carries private information. If the data is leaked, it will bring serious consequences. Therefore, researching and solving the data privacy protection problem in WSN is of great significance for the large-scale application of WSN [1].

The emerging blockchain technology has attracted extensive attention in WSN in recent years. Blockchain is famous for its decentralization, tamper-proof, and traceability. It can realize the safe sharing of ledger data in an open network environment without centralized

* Corresponding author at: School of Cyber Science and Engineering, Southeast University, Nanjing, 211189, China.
E-mail addresses: xyhu@njnet.edu.cn (X. Hu), xysong@njnet.edu.cn (X. Song), gcheng@njnet.edu.cn (G. Cheng), hwu@njnet.edu.cn (H. Wu), jgong@njnet.edu.cn (J. Gong).

https://doi.org/10.1016/j.comcom.2022.07.035
Received 28 February 2022; Received in revised form 28 June 2022; Accepted 20 July 2022
Available online 25 July 2022
0140-3664/© 2022 Published by Elsevier B.V.

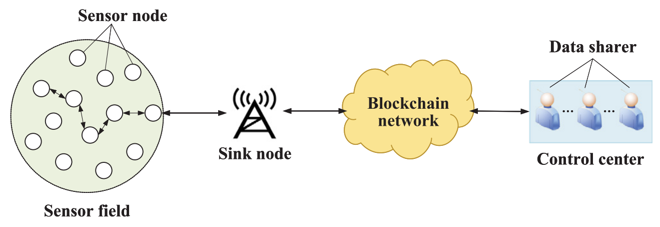


Fig. 1. Sensoring data sharing in WSN based on blockchain.

authorization [2]. Therefore, the blockchain-based approach empowers the transfer of collected information from intelligent devices to the information network, enabling secure and convenient sharing among users [3,4].

Fig. 1 is a schematic diagram of sensing data sharing in a WSN network based on blockchain. As illustrated, the sensor node monitors the target area, acquires data in the sensor network, and completes the communication with other sensor nodes. The sink node is responsible for summarizing the data sent by the sensor nodes, performing further data fusion and other operations, and finally uploading the processed data to the blockchain network. Finally, the multiple data sharers in the control center will analyze and process the data on the blockchain network. In particular, compared to the public blockchain (e.g., Bitcoin [5], Ethereum [6]) and private blockchain, consortium blockchain (e.g., Hyperledger Fabric [7]) is widely used as the underlying blockchain architecture for application development combined with WSN due to its auditability and high transaction rate.

The immutability feature over records on the blockchain help audit data access behavior or verify the integrity of shared data, improving the security of WSN. However, the blockchain cannot solve the privacy problems raised by data sharing. Instead, it is often used in conjunction with some other privacy-preserving technology [8,9]. For example, the data from the sensing device is encrypted by a key (named *sensorkey*) shared by the sensor node and the sink node and then sent to the blockchain network.

In this paper, we consider a scenario where the data collected from sensing devices is shared on the consortium blockchain in ciphertext to achieve the goal of privacy protection. The infrastructure in current large-scale WSN is usually not owned and managed by a single institution but by combining systems from multiple organizations and companies. Therefore, there will be various organizations in the control center, and the encrypted data on the blockchain can be authorized to these organizations for calculation. In this case, distributing the *sensorkey* to each organization requires separate encryption of the shared key using the organization's public key. The sink node needs to be online when each organization asks for the *sensorkey*, and one encryption of the *sensorkey* for each organization consumes precious resources. We abstract the organizations that request privacy-preserving sensing data into data sharers. Several data sharers represent different companies or interest groups and are identified in the initial WSN creation stage. These data sharers will form a stable group and perform corresponding sensing data acquisition and calculation.

Due to the stability of the data-sharing group and the need for multiple data sharers to efficiently obtain privacy-preserving sensing data, we apply group key agreement technology and propose a new system GSChain. GSChain represents group sharing of sensing data on consortium blockchain. In particular, multiple data sharers form a group and utilize the blockchain to generate a shared group encryption key and their respective group decryption keys. The *sensorkey* is encrypted only once by the group encryption key and stored on the consortium blockchain together with the encrypted private data. Date sharers then obtain the encrypted *sensorkey* from the blockchain and decrypt it with their respective group decryption key. The group key generation is involved only when the group forms or membership changes, which infrequently happens in this scenario.

Contributions: In this paper (an extended version of the work in [10]), we first analyze the privacy leakage problem when using the consortium blockchain in WSN to solve the data-sharing issue. Then we introduce the specific scenarios involved in privacy-preserving sensing data sharing. Access to encrypted data has shifted from multiple auditors to multiple data sharers. In addition, we supplement the research on the application of blockchain to provide data privacy protection in WSN and expand the discussion of private data sharing schemes on consortium blockchains.

In the current version of the paper, we update the details of the scheme used by GSChain for privacy-preserving sensing data sharing in the multi-data sharer scenario. We add a discussion (not present in [10]) on new data sharers wanting to obtain historical private data once delegated as group membership changes. We also discuss using Hyperledger Fabric's channel mechanism to provide better privacy and security for data governance. In addition, in the experimental part, we evaluate the situation of multiple data sharers joining or leaving the group simultaneously, the effect of the length of the *sensorkey* on the efficiency of sharing, and the quantitative analysis of the storage overhead caused by this scheme.

Organization: The remainder of this paper is organized as follows. First, we introduce the schemes of privacy-preserving data sharing in WSN and other areas based on consortium blockchain and the development of group key agreement technologies in Section 2. Section 3 elaborates on the design of GSChain for privacy-preserving sensing data sharing, followed by the experimental studies on the data-sharing performance in Section 4. Section 5 further analyzes the various overheads of our proposed GSChain and gives insights for optimization. Finally, we conclude the paper in Section 6.

2. Related work

This section briefly introduces the schemes of privacy-preserving data sharing on consortium blockchains in WSN and other IoT environments. We also present an overview of the primary and blockchain-based group key agreement protocols.

2.1. Privacy-preserving data sharing on consortium blockchains

Following the trend of using consortium blockchains for application development, the research community has explored several feasible solutions that provide privacy protection and data sharing. Shen et al. [11] constructed a dynamic and equitable incentive scheme for data sharing in multiple clouds using the consortium blockchain and Shapley value. Their solution also guarantees the security of data privacy. Zhang et al. [12] implemented a blockchain-based privacy-preserving Personal Health Information (PHI) sharing scheme for diagnosis improvements in e-Health systems. Their scheme consists of two chains. A private blockchain records the patient's original PHI (encrypted for security). The other consortium blockchain records indexes of PHI, thereby realizing data privacy protection and secure sharing. Yin et al. [13] proposed a model for hierarchical supervision and sharing of encrypted data on the chain called CROSS. By combining proxy re-encryption with a layered key distribution scheme, CROSS can effectively improve the privacy of the blockchain while maintaining security. Yuen proposed PAChain [14], which uses encryption technologies such as anonymous credentials and zero-knowledge range proof to protect and audit the privacy of sender, recipient, and transaction privacy on the consortium blockchain. In addition, PAChain was shown to integrate well with the Hyperledger Fabric framework. Malik et al. [15] provided a solution, PrivChain, for confidential transactions in the supply chain context. PrivChain uses Zero-Knowledge Range Proofs (ZKRP) to provide origin information without disclosing the exact location of a supply chain product. Xu et al. [16] developed zkrpChain, a privacy-preserving data auditing solution that supports standard-scope and arbitrary-scope zero-knowledge scope proofs for on-chain data.

Backed by blockchain technology, scholars also proposed data privacy protection and efficient data-sharing schemes for WSN and other IoT environments. Guerrero-Sánchez et al. [17] proposed a decentralized system based on blockchain and cryptographic tools to ensure the autonomy and security of the IoT system. They verified the accuracy of the scheme in a temperature and humidity sensing IoT-based WSN. Truong et al. [18] proposed SASH for data sharing under the IoT, which provides auditable access control policy updates and data owner remuneration. Xu et al. [19] proposed a blockchain-empowered differentially private data publishing framework for the area of Industrial Internet of Things (IIOT). The proposed framework can help realize privacy-preserving and fairness-guaranteed data sharing in IIoT applications. Ma et al. [20] proposed IoVChain, which realizes the secure sharing of Internet of Vehicles (IoV) data based on blockchain in the intelligent transportation sensor network. Han et al. [21] proposed a blockchain-based auditable access control system by applying blockchain technology and the ABAC model, which realizes the dynamic management of private data access control policy in service-centric IoT environments.

None of these solutions involves the efficiency of privacy-preserving sensing data with multiple data sharers, and we give an idea of using the group key agreement technology. The group key agreement protocol enables a set of data sharers to generate a shared group key for the subsequent privacy-preserving data sharing process. For example, the *sensorkey* used to encrypt the sensing data is encrypted only once by the group key and then distributed to multiple data sharers.

2.2. Basic and blockchain-based group key agreement protocols

Diffie et al. [22] firstly proposed a scheme for two communicators to establish the same key through negotiation. After that, Group Key Agreement (GKA) protocols suitable for multi-user scenarios have been proposed to provide secure communication between groups. Based on [22], Burmester et al. [23] proposed the first constant-round GKA protocol. The group key is generated by the joint contributions of each protocol participant. In order to ensure that only members with legitimate identities can participate in the process of group key agreement, Just et al. [24] proposed a four-round authenticable GKA protocol based on [23] and public-key cryptography. Subsequently, the ID-based cryptographic mechanism proposed by Shamir [25] was applied to GKA protocols to solve the certificate management problem under the public key cryptosystem. With the help of the ID-based cryptographic mechanism, each user generates a public key through his unique ID, which brings about the problem of password escrow. The user needs to trust Key Generation Center (KGC) to generate a private key fully. Therefore, Al-Riyami et al. [26] proposed a Certificateless Public-Key Cryptosystem (CL-PKC) in 2003, enabling the user and KGC to generate the key jointly. KGC cannot obtain the user's complete key pair, thus enhancing user privacy security. Heo et al. [27] proposed the first certificateless dynamic GKA protocol in 2007, which supports dynamic membership management and mutual authentication of members within a group, but lacks forward security. Teng et al. [28] constructed a certificateless GKA protocol with constant rounds and proved its security in a random oracle model. Based on [28], Semal et al. [29] proposed a certificateless authentication group key protocol for static groups, providing ideas for the research direction of providing trusted communication in untrusted Unmanned Aerial Vehicle (UAV) networks.

Symmetric group key means only group members can use the group key to encrypt information, limiting the network's openness. To address the problem, Wu et al. [30] proposed a single-round Asymmetric Group Key Agreement (AGKA) scheme based on the new concept of aggregatable signature broadcast for the first time in 2009. In this scheme, group members finally negotiate a shared group encryption key and privately held group decryption key, respectively. Zhang et al. [31] proposed an identity-based one-round AGKA protocol to provide membership authentication. In order to meet the needs of dynamic changes of group members, Zhao et al. [32] first proposed a three-round dynamic AGKA protocol, which can realize key negotiations between dynamic users in temporary groups. Chen et al. [33] proposed a dynamic AGKA scheme without certificate authentication and analyzed the security and efficiency of the scheme.

Due to the difficulty of tampering and traceability of blockchain technology, some researchers have noticed that blockchain technology can work for the secure information exchange and transmission between group members, making the process of group key agreement more flexible and efficient. Zhang et al. [34] realized the secure information exchange and transmission among group members through blockchain technology, and the proposed protocol supports anonymous identity authentication and traceability. Xu et al. [35] proposed a blockchain-based authentication and dynamic GKA protocol, which improves the efficiency of group member authentication and solves the problem of single node failure. Li et al. [36] proposed a blockchain-based group key distribution scheme in response to the loss of specific group broadcast messages in the Unmanned Aerial Vehicles Ad-Hoc Network (UAANET), which realizes the mutual recovery of lost group keys among nodes. Therefore, the group key agreement technology can be well integrated with the blockchain, which makes the group key agreement more flexible and practical and serves for the secure sharing of the privacy-preserving data on consortium blockchains.

3. Architecture design

This section presents the proposed GSChain enabling efficient privacy-preserving sensing data sharing in the multi-data sharer scenario. We first give an overview of the system structure. Next, we elaborate on the three parts of the process, including group key generation, privacy-preserving sensing data sharing, and the update of group keys for group dynamics. Finally, we conduct a theoretical analysis of the proposed scheme's efficiency, security, and privacy.

3.1. System overview

The structure of the sharing system for privacy-preserving sensing data is shown in Fig. 2. The sharing system consists of the following three layers.

Sensor Layer. The sensor node first collects sensing data, encrypt it with a specific *sensorkey*, and transmits it to the corresponding sink node. The sink node decrypts to obtain sensing data, performs further data aggregation and other operations, encrypts the resulting data with *sensorkey*, and then uploads it to the consortium blockchain network. There are multiple sensor nodes and sink nodes at the sensor layer.

Blockchain Layer. Sink nodes send the processed private sensing data to the blockchain layer as transactions. The transactions invoke the smart contracts and are stored on the blockchain. A set of access rules defined by the Access Control List (ACL) ensure that only authenticated nodes can join the blockchain network and read or write the data.

Application Layer. There are multiple data sharers from various organizations. They form a group and use the consortium blockchain to generate group keys. Data sharers can request the encrypted sensing data on the blockchain and get corresponding results. The administrator manages the entire blockchain network and has startup and installation privileges of smart contracts. The administrator is also responsible for generating system parameters (for group key generation) and maintaining groups of data sharers.

Fig. 3 shows the workflow for using the shared key to implement privacy protection in wireless sensor networks discussed in this paper. In order to avoid leakage of privacy during transmission, the sensor node should encrypt the sensing data using a shared *sensorkey*. The sink node performs corresponding decryption and aggregation operations on the acquired sensing data. Before submitting to the consortium

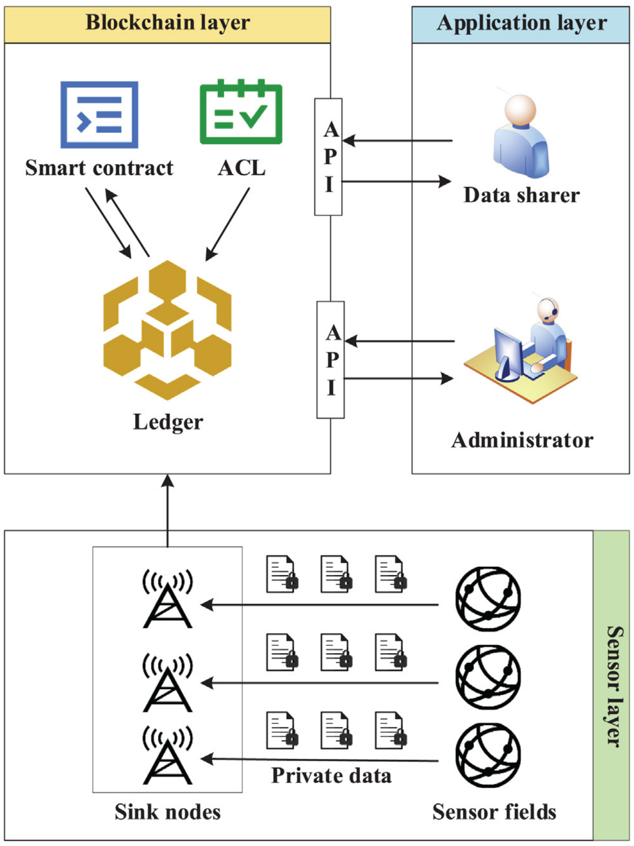


Fig. 2. System structure.

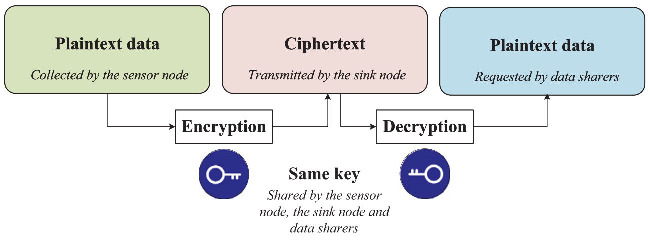

Plaintext data
Collected by the sensor node
Ciphertext
Transmitted by the sink node
Plaintext data
Requested by data sharers
Encryption
Decryption
Same key
Shared by the sensor node, the sink node and data sharers

Fig. 3. The workflow for using the shared key to implement privacy protection in WSN.

blockchain, the resulting data must be encrypted again with *sensorkey* to protect the privacy of sensing data sharing. Multiple data sharers in the control center need to obtain the sensing data in plaintext for subsequent calculation and processing. Therefore, they need to acquire the *sensorkey* of each privacy-preserving sensing data from the sink node. This paper omits the detailed discussion of the underlying WSN because the data-sharing step mainly involves the sink node, the data-sharing group, and the blockchain. In particular, the sink node undertakes the task of data transmission with high reliability and high performance. It is responsible for uploading the encrypted data collected from each sensor node to the blockchain. At the same time, it needs to respond to the authorization demand of the data-sharing group for specific encrypted data.

3.2. Group key generation

The group key generation scheme in GSChain is based on the proposal in [33]. Our amelioration is that the generated information during the negotiation process of each group member is deeply coupled with

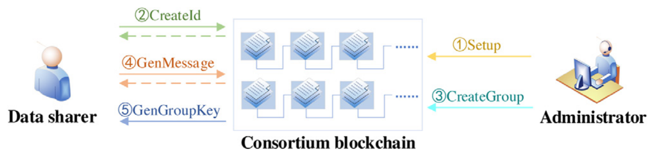


Fig. 4. The scheme of group key generation.

the blockchain while allowing personal private data to be stored locally. Data sharers generate a shared group encryption key in this process, and each owns a group decryption key. In addition, the generated group encryption key will be broadcast on the blockchain network for sink nodes to encrypt the *sensorkey*.

Assuming that n data sharers need to generate group encryption and decryption keys to obtain the ciphertext of the specific sensing data (submitted to the consortium blockchain as a transaction), a group $A = (u_1, u_2, \dots, u_n)$ is formed. The scheme comprises polynomial-time algorithms $\mathrm{I} = (Setup, CreateId, CreateGroup, GenMessage, GenGroupKey)$.

As shown in Fig. 4, each data sharer needs to interact with the blockchain to generate personally identifiable information, broadcast messages, and more. In addition, the administrator starts the blockchain network, generates system parameters, and creates groups of data sharers.

(1) Setup$(\lambda) \to (params)$. With a security parameter λ, this algorithm generates a list of public parameters $params$ which are stored on the blockchain and made available to anyone. Note that the administrator executes this algorithm only once for the blockchain.

(2) CreateId$(params, x) \to (id, (PK, SK))$. Given public parameters $params$ and the random number x, this algorithm initializes an identification id and key pair (PK, SK) for a user u, where id represents the unique identification of the user in the blockchain network. *Note that the user here refers to a data sharer.* (PK, SK) is identified as the user's public and private key pair and is expressed as follows:

$$PK = (Q, P), SK = (D, x) \tag{1}$$

where Q, P, and D are respectively calculated by id, x, and Q combined with $params$. The algorithm is executed only once for each user, and the id will be used to uniquely identify all behaviors of the user in the blockchain network. The user's private key SK will be stored locally, while the public information submitted to the blockchain is as follows:

$$Userinfo = (id, PK) \tag{2}$$

(3) CreateGroup$(A, \theta) \to (Group_A)$. Given a set of users A and session state information θ, this algorithm generates a group that needs to obtain group keys. The specific content of $Group_A$ is shown as follows:

$$Group_A = ((id_1, 1), (id_2, 2), \dots, (id_n, n)) \tag{3}$$

Take u_i as an example, the information recorded in $Group_A$ is (id_i, i), where id_i denotes the value of id for u_i, and i denotes the index value of the user in the group. $Group_A$ will also be kept on the blockchain as public information. Note that the execution of this algorithm is related to the dynamics of the data-sharing group. Every time the group changes, the algorithm needs to be executed once.

(4) GenMessage$(params, \theta, Group_A, u_i, r_i) \to (Message_{Ai}, s_{i,i})$. This algorithm enables a user of the group A (say u_i) to generate broadcast messages $Message_{Ai}$ and its private value $s_{i,i}$ in the group key negotiation process for the set A. The specific content of $Message_{Ai}$ is as follows:

$$Message_{Ai} = (R_i, s_{i,1}, s_{i,2}, \dots, s_{i,n(i \neq j)}) \tag{4}$$

The user u_i first selects the random number r_i and then generates R_i using r_i and $params$. Subsequently, u_i obtains the indexes of other users in the group according to the group information $Group_A$ and combines

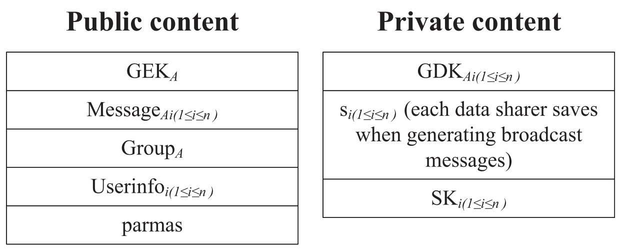

Public content
Private content
GEK_A
$Message_{Ai(1\leq i\leq n)}$
$Group_A$
$Userinfo_{i(1\leq i\leq n)}$
parmas
$GDK_{Ai(1\leq i\leq n)}$
$s_{i(1\leq i\leq n)}$ (each data sharer saves when generating broadcast messages)
$SK_{i(1\leq i\leq n)}$

Fig. 5. Public and private contents generated during group key generation.

$params$, θ, and its own information $(id_i, (PK_i, SK_i))$ to calculate messages $s_{i,j(1\leqslant j\leqslant n)}$ for all users. Among them, $s_{i,i}$ will be stored locally by u_i as a private value. Other contents make up $Message_{Ai}$ and will be provided to all members in the group as public information.

All members in group A need to perform this step and submit the $Message_{Ai(1\leqslant i\leqslant n)}$ to the consortium blockchain. Due to access restrictions and fast transaction processing speed, the consortium blockchain can effectively protect the security of group communication content and improve the performance of group key negotiation during this process.

(5) GenGroupKey($params, Group_A, Userinfo_A, Message_A, \quad u_i$) $\rightarrow$ (GEK_{Ai}, GDK_{Ai}). This algorithm enables a user of group A (say u_i) to generate group encryption and decryption keys. $Userinfo_A$ and $Message_A$ denote a collection of public information and broadcast messages of all users in group A. The data sharer u_i first composes a key generation matrix according to $Message_A$ and then verifies the integrity and correctness of the obtained broadcast messages through $Userinfo_A$. If the verification is correct, the group encryption key GEK_{Ai} and the group decryption key GDK_{Ai} are calculated according to the key generation matrix and private information such as SK_i and $s_{i,i}$.

For all users of group A, their group encryption key $GEK_{Ai(1\leqslant i\leqslant n)}$ is the same, but the group decryption key $GDK_{Ai(1\leqslant i\leqslant n)}$ is different.

Fig. 5 shows the public content stored on the consortium blockchain during the group key generation process and the private value stored locally by each data sharer.

3.3. The process of privacy-preserving sensing data sharing

The privacy-preserving sensing data sharing process in GSChain is shown in Fig. 6. With the help of the group key generation process, data sharers first generate a group key pair through the consortium blockchain and broadcast the group encryption key to all sink nodes in the WSN. The sensing data is encrypted with the $sensorkey$, and this $sensorkey$ is encrypted with the group encryption key. Each data sharer does not need to issue an authorization request to the sink node separately in our solution. Instead, the encrypted result of the sensing data and the $sensorkey$ will be packaged into a single transaction, which will be uploaded to the consortium blockchain by the sink node. Then, each data sharer can decrypt the definitive sensing data according to his group decryption key and the transaction records on the blockchain.

Sink nodes also need to use the $createId()$ algorithm to generate their identification information and submit transactions with private data after obtaining the current group encryption key. The data sharer can decrypt the $sensorkey$ with its group decryption key and then obtain the plaintext of private sensing data. The data sharer can use the acquired result for subsequent processing and computation. The relative steps are shown in Fig. 7.

The whole privacy-preserving sensing data sharing process in GSChain can be realized by the following algorithms Π = ($GenSecretSensorData, GenCipher, DecodeCipher$). Note that the group encryption keys of all data sharers in group A are uniformly denoted as GEK_A due to consistency.

(1) GenSecretSensorData($data, sensornode, sensorkey$) $\rightarrow$ ($SecretSensorData$). Given the plaintext sensing data $data$ and the shared $sensorkey$, this algorithm enables the sensor node to use the given $sensorkey$ to encrypt the $data$ and get the ciphertext. The $SecretSensorData$ is then transmitted to the corresponding sink node. The specific contents of $SecretSensorData$ are as follows:

$$SecretSensorData = (sid, encryptedresult) \tag{5}$$

where sid is used to uniquely identify the encrypted sensing data, and $encryptedresult$ denotes the specific content after encryption.

(2) GenCipher($params, sid, sensorkey, sinknode, GEK_A$) $\rightarrow$ ($Cipher$). This algorithm enables sink nodes to use the given group encryption key GEK_A to encrypt the $sensorkey$ and get the corresponding ciphertext. The specific contents of $Cipher$ are as follows:

$$Cipher = (sid, ciphertext, time, version) \tag{6}$$

where $ciphertext$ denotes the specific content after encryption, $time$ denotes the current time of the data sharing, and $version$ denotes the version information of the group encryption key used. Note that $version$ is prepared for the situation when the group changes dynamically and the details will be discussed in detail in Section 3.4. The $Cipher$ only needs to be generated once under the action of the group encryption key and will be stored together with the $SecretSensorData$ on the blockchain.

(3) DecodeCipher($Cipher, GDK_{Ai}, u_i$) $\rightarrow$ ($sensorkey$). This algorithm enables the data sharer (say u_i) to obtain the desired $sensorkey$ using its own group decryption key.

It needs to be pointed out that since the encryption result of the $sensorkey$ is stored on the blockchain along with the $SecretSensorData$, sink nodes do not need to be online in real time to respond to data sharing requirements. The data sharer can directly combine the on-chain data and its group decryption key to obtain the entire content of the sensing data in plaintext. Furthermore, since only members with the group decryption key can correctly decrypt and obtain the $sensorkey$, even if the encryption result is stored as public content on the blockchain, the privacy of the sensing data would not be undermined.

3.4. Group key update when group membership changes

The members of the data-sharing group will usually remain stable for some time, but it will also change with the specific needs. New organizations will join the original WSN to perform new functions for various needs. In such a case, the group key needs to be updated. Note that this situation is relatively rare.

The group key update scheme for group dynamics is similar to the content described in Section 3.2, except that the system parameters do not need to be regenerated. Assuming that a new data sharer (say u_{n+1}) joins group A, the group becomes $B = (u_1, u_2, \dots, u_n, u_{n+1})$. First, the u_{n+1} will generate its own identification and public and private key information and save the public information on the blockchain. $Group_B$ is also generated as a new group content according to the new set of users in B. Then, each member in group B regenerates the broadcast message $Message_{Bi}$ and the private value $s_{i,i}$. Finally, the new group encryption and decryption keys of each data sharer (GEK_{Bi}, GDK_{Bi}) will be generated by combining $params$, $Group_B$, $Userinfo_B$, $Message_B$ and their respective private information.

Assuming that a member (say u_i) leaves the group A, the group changes to $C = (u_1, u_2, \dots, u_{i-1}, u_{i+1}, \dots, u_n)$. Similarly, $Group_C$ is generated, and members of group C regenerate contents such as broadcast messages. These will be collectively used as the input of the algorithm $GenGroupKey$, and finally each group member will get a new group key pair (GEK_{Ci}, GDK_{Ci}).

The $version$ field in $Cipher$ is used to mark the corresponding group encryption key information. Therefore, each data sharer can know which group decryption key is used for decryption. Fig. 8 displays the data stored on the consortium blockchain when the data-sharing group changes. For example, the newly added user (say u_{n+1}) can decrypt the

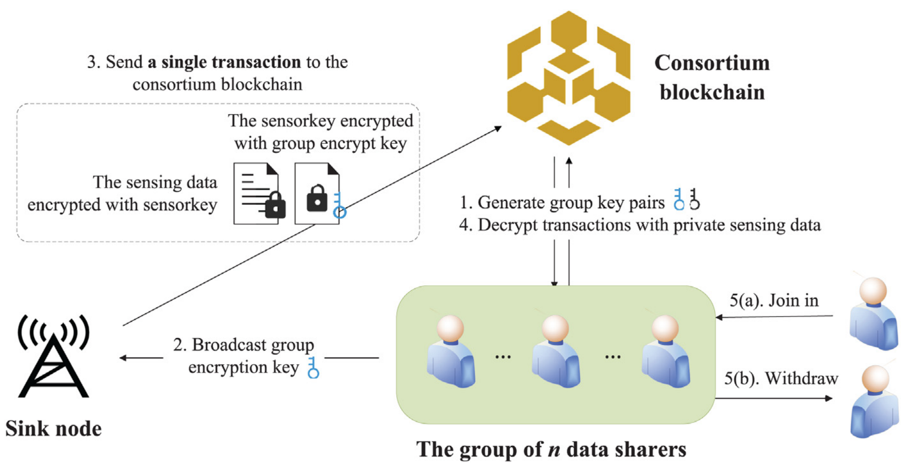

3. Send a single transaction to the consortium blockchain
The sensorkey encrypted with group encrypt key
The sensing data encrypted with sensorkey
Consortium blockchain
1. Generate group key pairs
4. Decrypt transactions with private sensing data
5(a). Join in
5(b). Withdraw
2. Broadcast group encryption key
Sink node
The group of n data sharers

Fig. 6. The privacy-preserving sensing data sharing process in GSChain.

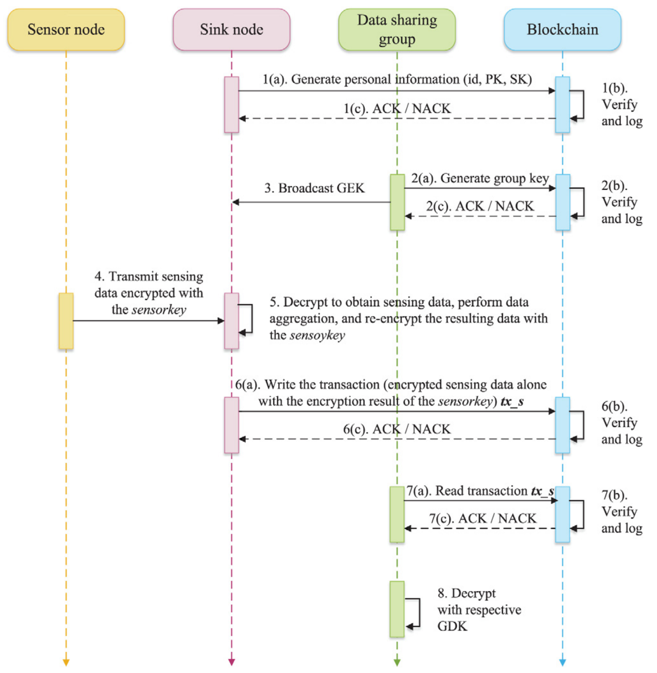

Sensor node
Sink node
Data sharing group
Blockchain
1(a). Generate personal information (id, PK, SK)
1(b). Verify and log
1(c). ACK / NACK
3. Broadcast GEK
2(a). Generate group key
2(b). Verify and log
2(c). ACK / NACK
4. Transmit sensing data encrypted with the sensorkey
5. Decrypt to obtain sensing data, perform data aggregation, and re-encrypt the resulting data with the sensoykey
6(a). Write the transaction (encrypted sensing data alone with the encryption result of the sensorkey) tx_s
6(b). Verify and log
6(c). ACK / NACK
7(a). Read transaction tx_s
7(b). Verify and log
7(c). ACK / NACK
8. Decrypt with respective GDK

Fig. 7. The steps of transaction sharing in GSChain.

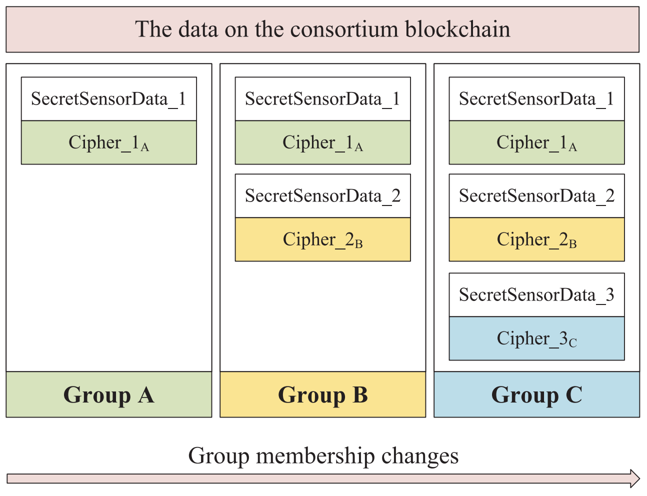


Fig. 8. The data stored on the blockchain when the group changes.

wireless network data (*SecretSensorData*_2) stored on the blockchain but cannot decrypt the old data (*SecretSensorData*_1). The exiting user (say u_i) cannot decrypt the newly stored data (*SecretSensorData*_3). Still, the historical data (*SecretSensorData*_1 and *SecretSensorData*_2) can be interpreted by the corresponding version of the group decryption key stored locally. If there is a need to restrict the exiting data sharer's access to historical data, ACL can serve to limit its access to the blockchain network.

There is a realistic requirement that the newly added user u_{n+1} needs to read the historical encrypted data (*SecretSensorData*_1). One feasible option is that the sink node encrypts the historical data with the new group encryption key and uploads it to the blockchain. This approach will cause ample data redundancy to the blockchain network, especially when there is a lot of historical data. Our solution is to put a fixed user's historical group decryption key in the data-sharing group on the chain and use the channel mechanism of consortium blockchains to protect the privacy of the group decryption key. There will be a permanent user named u_0 in the data-sharing group. When the group membership changes, u_0 will upload the old group key pair in plaintext to the specific channel of the consortium blockchain. In this way, sink nodes do not need to repeatedly respond to new data sharers' sharing demands for historical privacy sensing data, thus avoiding energy consumption and data redundancy. The system administrator can perform this role. Its function is not to authorize private transactions but to provide the group key information of all historical versions. Then if u_{n+1} wants to obtain the *SecretSensorData*_1, he can obtain the group key of the previous version of group *A* from the blockchain and then proceed to the decryption step.

Suppose u_0's historical group decryption key is disclosed on the consortium blockchain without taking other measures. In that case, all users can see this content, thus disclosing the historical privacy-preserving sensing data. Since GSChain is deployed on Hyperledger Fabric, we can use the channel mechanism to make the private sensing data invisible to other non-data sharers. A channel is a dedicated "subnet" for communication between two or more specific network members constructed based on data isolation and confidentiality in Fabric. As Fig. 9 shows, all data sharers can join *Channel X* for group key generation and update operations. In addition, the group key information uploaded each time by u_0 is also propagated only within this channel. *Channel Y* is used exclusively for sink nodes and data-sharing groups for privacy-preserving sensing data sharing operations. Using the multi-channel feature of Hyperledger Fabric, data sharers can organize their data under the same channel, while organizations

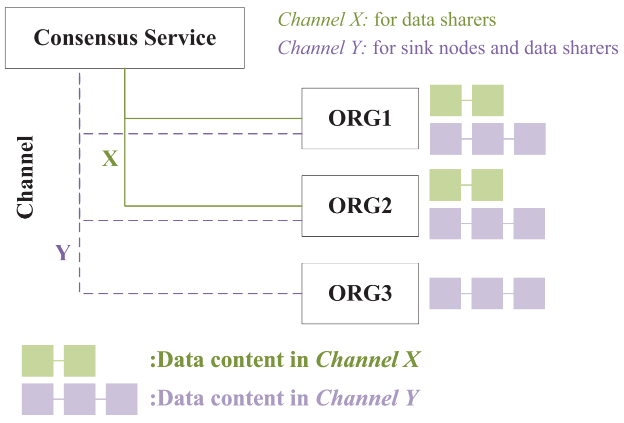

Consensus Service
Channel X: for data sharers
Channel Y: for sink nodes and data sharers
Channel
X
Y
ORG1
ORG2
ORG3
:Data content in Channel X
:Data content in Channel Y

Fig. 9. The channel mechanism used in the scheme.

outside the channel cannot view data details. In this way, other non-data sharers will not obtain any information about the group decryption key. In fact, if u_0 puts $sensorkeys$ of all historical data into this channel, it can achieve the same effect as placing the group decryption key of the historical version into this channel. However, if a large number of $sensorkeys$ are stored in the channel, it will cause a heavy data load burden to the resource-intensive channel, thereby affecting the performance of the entire blockchain system.

3.5. *Theoretical analysis*

3.5.1. *Efficiency analysis*

The private sensing data sharing scheme in GSChain allows the $sensorkey$ to be encrypted only once using the group encryption key of the data-sharing group. The encryption results are stored on the blockchain along with the private sensing data, allowing data sharers to obtain the data collected by sensors on-demand without asking sink nodes for the $sensorkey$. If the group key agreement method is not used, the $sensorkey$ needs to be encrypted and distributed n times using the public keys of n data sharers. Meanwhile, sink nodes need to remain online to respond to each data sharer's request for particular sensing data. When a new data sharer wants to obtain historical encrypted data, our solution does not require sink nodes to respond to repeated data-sharing requests. Instead, the new data sharer reads the $Cipher_A$ and the (GEK_{A0}, GDK_{A0}) (protected by the channel mechanism) on the blockchain to get the plaintext, saving time and storage overhead of repeated encryption and uploading of the historical sensing data to the blockchain.

In brief, the proposed system in this paper improves the sharing efficiency of privacy-preserving sensing data on consortium blockchains by reducing the number of encryptions while alleviating the online requirements for sink nodes.

3.5.2. *Privacy and security analysis*

Since the asymmetric group key agreement scheme is used, any entity in the network can use the group encryption key to encrypt and send messages. Only members of the data-sharing group can decrypt the corresponding content, thus ensuring the privacy of sensing data. The use of channels further ensures that the messages of data sharers are confidential to sink nodes. In addition, the private information of each data sharer is stored locally, which indicates that data sharers cannot calculate what the group decryption keys of other group members are. Privacy among data sharers has also been effectively protected.

The security challenge of this solution is the leakage of private data. One possible situation is that members of the data-sharing group are malicious or attacked, which results in the exposition of group decryption keys and thus the disclosure of privacy-preserving sensing data. The loss caused by this is consistent with the disclosure of private keys of data sharers in the traditional scheme. Therefore, the security provided by GSChain is the same as the traditional transaction-sharing solution.

Another possible situation is that members who have left the group can still use the old group decryption key to decrypt newly submitted transactions containing private data. In this regard, GSChain requires that the data-sharing group's group encryption and decryption keys to be regenerated whenever the group members change dynamically. While regenerating group keys, the submission of transactions containing private data will be prohibited. This restriction ensures that the privacy-preserving sensing data is always shared with the latest version of the group key pairs under concurrent conditions.

4. Performance evaluation

This section provides a quantitative performance evaluation of our proposed GSChain. We mainly evaluate the impact of GSChain on the efficiency of privacy-preserving sensing data sharing in the multi-data sharer scenario and the time and storage overhead brought by the implementation of the scheme.

4.1. *Experimental setup*

We use the Type A curve in the jPBC library (Java Pairing-Based Cryptography Library)[1] to provide cryptographic support for group key generation. The consortium blockchain framework we adopt is Hyperledger Fabric,[2] an open-source blockchain platform managed by the Linux Foundation. We built a simple blockchain network based on Fabric v1.4, including two peer nodes and an orderer node, and all nodes are in the same channel. This channel is only joined and communicated by data sharers. The experiments for various scenarios are run on a 3.20 GHz CPU with a RAM of 16.0 GB.

Methodology. We evaluate the efficiency improvement of transaction sharing for privacy-preserving sensing data produced by adopting the group key agreement. Before experimenting, we initialize the system parameters and put the public parameters on the consortium blockchain (indicating the execution of the algorithm $Setup()$). The solution requires a total of four steps: the generation of the data-sharing group (indicating the execution of the algorithm $CreateGroup()$), the generation of basic information about data sharers (indicating the execution of the algorithm $CreateId()$), the generation of group encryption and decryption key (indicating the execution of the algorithm $GenGroupKey()$), and the encryption and propagation of the $sensorkey$ (indicating the execution of the algorithms of $GenCipher()$ and $DecodeCipher()$). Since the generation of encrypted sensing data is not the focus of this paper, we omit the execution of the algorithm $GenSecretSensorData()$. The third and fourth steps of the scheme without the group key agreement are different from that of GSChain. Specifically, the third step is omitted; in the fourth step, the sensorkey is encrypted and transmitted by the public key of the data sharer, and the data sharer uses its private key to decrypt it.

To evaluate the practicality of the proposed GSChain, we also measure the time it takes the group to update the group encryption and decryption keys for group dynamics, indicating how soon the system can resume regular operations.

Performance Metrics. The metrics to quantify the effectiveness of the system are mainly:

- **The time for group key generation**: the average time it takes a data sharer to generate group encryption and decryption key.
- **The storage overhead for group key generation**: the average storage overhead it takes the data-sharing group to generate group encryption and decryption key in the channel.

[1] http://gas.dia.unisa.it/projects/jpbc/.
[2] https://www.hyperledger.org/use/fabric.

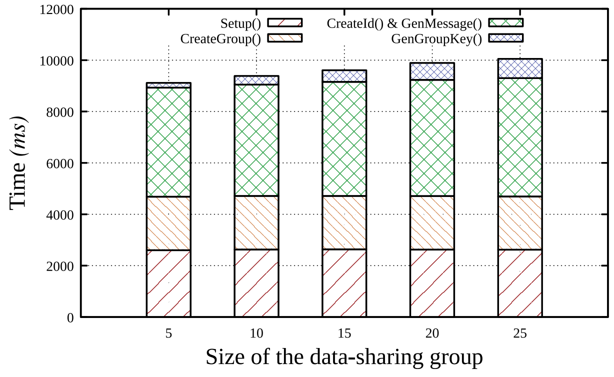


Fig. 10. The average time to generate group keys versus the group size.

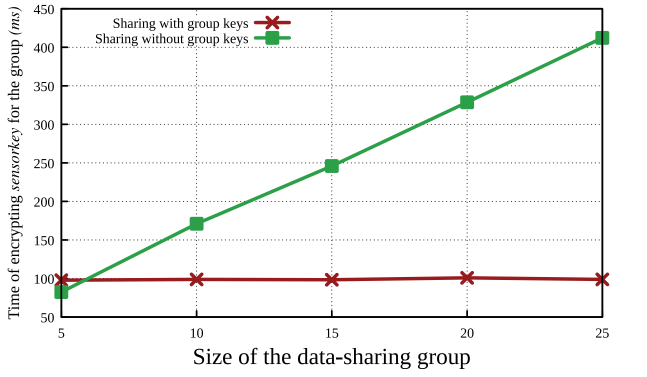


Fig. 12. The average time of encrypting *sensorkey* for the data-sharing group with and without group keys versus the group size.

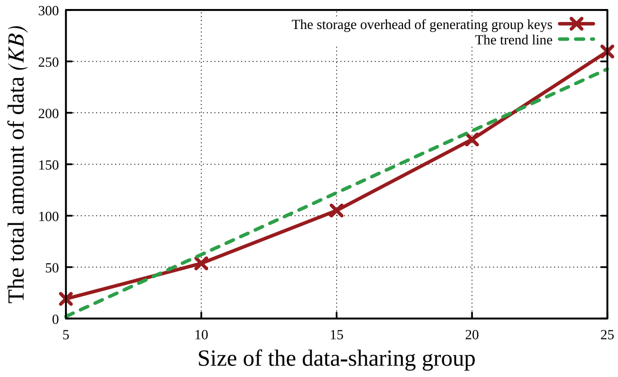


Fig. 11. The total amount of data in the channel when generating group keys versus the group size.

- **The time for privacy-preserving sensing data sharing**: the average time it takes the data-sharing group to get the *sensorkey* of the privacy-preserving sensing data.
- **The time for group key update**: the average time it takes a data sharer to update the group encryption and decryption key when the group membership changes.

4.2. Experimental results

4.2.1. The time and storage overhead for group key generation

The members of the data-sharing group obtain the group key through a key generation matrix, and the dimensions of rows and columns of the matrix are related to the number of data sharers. Therefore, the size of the data-sharing group directly affects the speed of group key generation. In this simulation, the number of data sharers varies between 5 to 25, and other parameters remain unchanged.

Fig. 10 illustrates the average time of different steps to generate group keys for different sizes of data-sharing groups. Note that steps $CreateId()$, $GenMessage()$ and $GenGroupKey()$ calculate the average time taken by a single data sharer to carry out the relevant operations. As displayed, the generation time of system parameters remains basically unchanged, while the time for a single data sharer to generate identification information and broadcast messages shows a slight increase as the group expands. The generation of broadcast messages mainly causes this increase. The more data sharers included in the group, the more broadcast information needs to be calculated, thus increasing the time overhead of this part. It can also explain the increase in the time it takes a single data sharer to generate group encryption and decryption keys. The larger the data-sharing group size, the longer the average time to generate group keys.

Fig. 11 displays the variation of the total amount of data generated in the channel for the data-sharing group to generate group keys. We use linear functions to fit the storage overhead of generating group keys for data-sharing groups of different sizes. The specific linear function is $y = 12.025x - 58.085$. The figure shows a significant growth trend. The increase is mainly caused by uploading user information and broadcast messages. The more data sharers there are, the more user information and broadcast messages are generated for the group, resulting in increased storage overhead on the blockchain. Furthermore, the trend line indicates that when the number of group members increases to 100, the total amount of data generated in the channel for group key generation is about 1 MB. In reality, the size of the data-sharing group should be much smaller than that. Therefore, the storage overhead generated by the scheme in this paper is tolerable.

4.2.2. The time for privacy-preserving sensing data sharing

Subsequently, we measure the time efficiency improvement contributed by adopting the group key agreement to the privacy-preserving sensing data sharing. Since all data sharers can obtain the *sensorkey* simultaneously, the group key mechanism mainly affects the sink node encrypting *sensorkey* for the whole data-sharing group. The size of the private sensing data encrypted by the *sensorkey* obviously directly impacts the size of the transaction. However, the focus of our scheme is to use the group key to encrypt and decrypt the *sensorkey*. Therefore, we consider the encryption results of the *sensorkey* as a single transaction and submit it to the blockchain for simplicity. Besides, to show the influence of the group key mechanism on privacy-preserving sensing data sharing more clearly, we only measure the time required by the sink node to encrypt the *sensorkey* for the data-sharing group. In this simulation, we assume that the length of the *sensorkey* is 128 bits, and the number of data sharers varies from 5 to 25.

Fig. 12 illustrates the average time of privacy-preserving sensing data sharing with and without group keys. When the size of the group is 5, the time for sharing private sensing data with group keys takes longer than that without group keys. The reason is that encryption using the group key is more complicated than using the data sharer's public key alone. When the group of data sharers becomes larger and larger, the advantage of using the group key mechanism will become more prominent. The variation trend of the experimental results is consistent with the efficiency analysis results of GSChain. Since the scheme under the group key mechanism only requires the sink node to encrypt the *sensorkey* once, the time for sharing privacy-preserving sensing data remains unchanged regardless of group size. On the contrary, the time overhead for sharing without group keys is proportional to the size of the data-sharing group. The reason is that the sink node needs to encrypt the *sensorkey* as often as there are members in the data-sharing group.

We also measure the impact of the length of the *sensorkey* on the time for encryption with and without group keys. In this simulation,

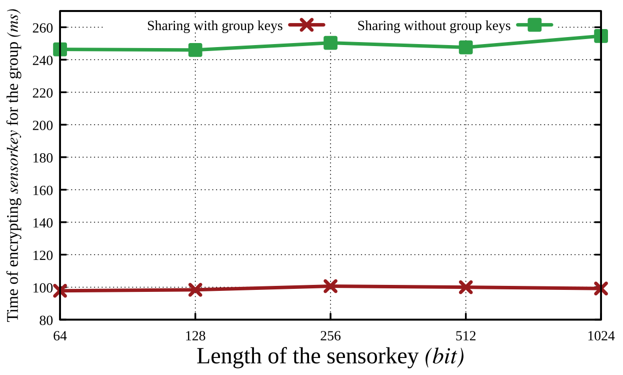


Fig. 13. The average time of encrypting *sensorkey* with and without group keys versus the *sensorkey* length.

we assume that the number of data sharers is 15, and the length of the *sensorkey* varies from 64 bits to 1024 bits.

Fig. 13 shows that the length of *sensorkey* has no significant effect in the two cases, i.e., privacy-preserving data sharing with or without the group key agreement. As the key length increases, the time it takes to encrypt the *sensorkey* using the group key mechanism changes marginally. Meanwhile, the privacy-preserving data sharing with the group key agreement outperforms that without the group key agreement in efficiency, consistent with the previous analyses. However, the size of the generated *Cipher* increases from 1.3 kB to 3.2 kB when the *sensorkey* increases from 64 bits to 1024 bits. Therefore, the size of the *Cipher* and the length of the *sensorkey* are positively correlated. Some other factors, such as the size of the consortium blockchain and the consensus algorithm used, have more influence on the speed of transaction submission, which goes beyond the scope of the paper.

4.2.3. The time for group key update

We evaluate the time it takes new group members to update group encryption and decryption keys when the original group has a data sharer joining and withdrawing. In this experiment, we assume that the original size of the data-sharing group varies from 5 to 25 and that one member joins or quits each time.

Fig. 14 depicts the time to update the group encryption and decryption keys when a member joins or leaves a group of different sizes. As illustrated, updating the group key pairs takes longer when a new member joins than when a member exits. The reason is that there is an additional step to generate identification information and public and private keys for the new member in the joining process. The average time spent on other steps is basically the same. Existing users do not need to regenerate identification information and public and private keys. The reason can also explain that when the data-sharing group changes, the time to update group keys will be shorter than the time it takes for the original group to generate them.

We also conduct additional experiments on the effect of simultaneous joining or withdrawing from multiple data sharers on the group key update when the number of the original group members is 15. The results show that if the number of group members remains unchanged when multiple members join or leave, the time for the new group to regenerate the group key remains the same. Therefore, we can conclude that the time of updating the group encryption and decryption keys is still proportional to the group size.

When the group changes, u_0 will upload its group decryption key to the consortium blockchain. The new data sharer needs to read the specific transaction and the group decryption key of a specific version to decrypt the desired sensing data. It should be pointed out that the storage overhead occupied by the group decryption keys uploaded to the blockchain should be relatively tiny due to the low frequency of group membership changing.

Table 1
The functions executed and the time consumed by the data-sharing group for privacy-preserving sensing data sharing.

The name of functions to execute	Time consuming (ms)
Setup()	2612.2
CreateGroup()	2077.5
CreateId()	2464
GenMessage()	2544.833
GenGroupKey()	546
GenCipher()	25
DecodeCipher()	22

Table 1 summarizes the time the system takes to perform each step when the size of the data-sharing group is 15. The default *sensorkey* length is 128 bits. The execution of the first five functions includes uploading the corresponding data to the chain and reading the corresponding data from the chain, which takes a long time to interact with the blockchain. Note that the time to execute these functions is related to the capacity of the working machine.

5. Discussion

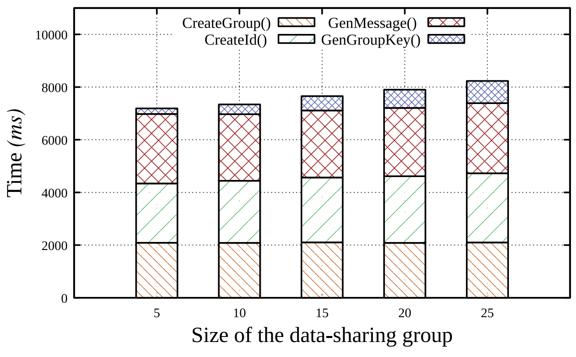
The proposed GSChain mainly helps improve the sharing efficiency of privacy-preserving sensing data on consortium blockchains. It also reduces the real-time requirements for sink nodes. In the deployment and implementation process, two aspects of overhead have increased. One is the time overhead of group key generation. It can be obtained from experiments that the time to generate group keys varies with the size of the data-sharing group. In the scenario discussed in this paper, the frequency of group membership changes is low. The time overhead for group key generation is usually one-time, which is tolerated. Even after the subsequent group membership changes, the update of group keys will bring some additional overhead, which is relatively small.

Another part of the overhead comes from the storage of *Cipher*. Under the traditional scheme, the encrypted result of the *sensorkey* is unicast sent to each data sharer by sink nodes and does not need to be stored on the blockchain. In GSChain, the *sensorkey* encrypted with the group key is stored on the blockchain, which means an increase in storage overhead. Nevertheless, our approach eliminates the need for sink nodes to frequently respond to data-sharing requirements for the same privacy-preserving sensing data.

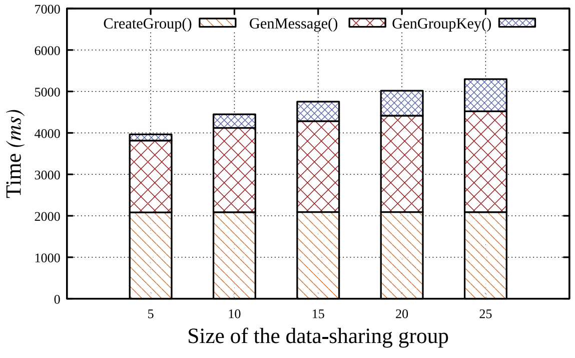
If the storage overhead is more of a concern, we propose that each sink node uploads the encrypted *sensorkey* as a transaction to the consortium blockchain only when the *sensorkey* is used for the first time. The subsequent sensing data encrypted by the *sensorkey* would specify the transaction for the encrypted *sensorkey* when stored on the consortium blockchain. In this way, we can save the repeat uploading and storage of the encrypted *sensorkey* to the consortium blockchain. Then the storage overhead is only slightly increased over the traditional practices.

There is still room for system optimization. First, GSChain tacitly assumes that data sharers have equal authority over blockchain networks, meaning they can see all encrypted transaction data. However, data sharers may have different powers in actual deployment and are individually asked to acquire specific privacy-preserving sensing data. For example, encrypted sensing data *SecretSensorData_1* are visible only to group *A*. The other set of crypto data, *SecretSensorData_2*, is visible only to group *B*. If the collaborative data-sharing is performed again, different group keys must be generated for different groups. The change of group members will also bring more consumption to the system. Therefore, it is worth considering how to efficiently share private-preserving sensing data in such a complex scenario.

The channel mechanism might work, but multiple channels are also resource-intensive. In addition, this paper does not discuss the underlying details of WSN, such as data aggregation of sink nodes, nor the operations of data sharers in the control center after obtaining



(a) The joining of a new data sharer



(b) The withdrawal of a data sharer

Fig. 14. Simulation results of the average time to update group keys for group dynamics versus the group size.

specific sensing data. Data sharers' demands may end with shared access to privacy-preserving sensing data. However, data sharers are more likely to use the acquired sensing data as input for their specific functions to perform different actions. The feedback from multiple data sharers on the sensing data can also be used to manage WSN.

6. Conclusion & Future work

This work proposes GSChain and establishes an efficient scheme for sharing privacy-preserving sensing data on consortium blockchain under the scenario of multiple data sharers in WSN. GSChain combines blockchain and dynamic asymmetric group key agreement technology. It treats data sharers as a group and generates a shared group encryption key and a group decryption key for each data sharer to efficiently distribute the *sensorkey* that encrypts the private sensing data on the consortium blockchain. GSChain is suitable for integrating blockchain technology into the WSN ecosystem to improve the sharing efficiency of privacy-preserving sensing data and reduce the real-time demand of sink nodes. We implement our scheme on the Hyperledger Fabric and validate its improvement of the privacy-preserving sensing data-sharing efficiency through experiments. We evaluate the system availability in the event of group membership changes. We also discuss the various overheads and possible system optimizations in detail. Our next step is to refine GSChain's practice of effectively sharing privacy-preserving sensing data in response to data sharers with different power levels and expand the specific operations of data sharers after obtaining a large amount of sensing data.

CRediT authorship contribution statement

Xiaoyan Hu: Conceptualization, Methodology, Investigation, Formal analysis, Supervision, Writing – review & editing. **Xiaoyi Song:** Software, Data curation, Methodology, Validation, Writing – review & editing. **Guang Cheng:** Funding acquisition, Resources, Writing – review & editing. **Hua Wu:** Visualization, Writing – review & editing. **Jian Gong:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to acknowledge the editors and anonymous reviewers. This research was supported in part by the Future Network Scientific Research Fund Project, China under Grant FNSRFP-2021-YB-01, in part by the Nanjing Social Science Foundation Key Project, China under Grant 20ZY06, in part by the Foundation of Key Laboratory of National Defense Science and Technology Project, China under Grant 6142103200101, in part by the Jiangsu Science and Technology Project, China under Grant BY2020404, in part by the National Natural Science Foundation of China Project under Grant 61602114, and in part by the Zhishan Youth Scholar Program of Southeast University, China under Grant 2242021R41121.

References

[1] Y. Yu, W. Peng, J. Lu, Wireless network security game based on conditional privacy policy, Comput. Commun. 184 (2022) 96–106.
[2] Y. Yuan, F. Wang, Blockchain: The state of the art and future trends, Acta Automat. Sinica 42 (2016) 481–494.
[3] X. Wang, X. Zha, W. Ni, R.P. Liu, Y.J. Guo, X. Niu, K. Zheng, Survey on blockchain for internet of things, Comput. Commun. 136 (2019) 10–29.
[4] M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities, IEEE Internet Things J. 6 (5) (2019) 7702–7712.
[5] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, in: Decentralized Business Review, 2008, p. 21260.
[6] D.D. Wood, Ethereum: A secure decentralised generalised transaction ledger, Ethereum Proj. Yellow Pap. 151 (2014) (2014) 1–32.
[7] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A.D. Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K.A. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolic, S.W. Cocco, J. Yellick, Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the Thirteenth EuroSys Conference, 2018, pp. 1–15.
[8] H. Lin, S. Garg, J. Hu, X. Wang, M.J. Piran, M.S. Hossain, Privacy-enhanced data fusion for COVID-19 applications in intelligent internet of medical things, IEEE Internet Things J. 8 (21) (2020) 15683–15693.
[9] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT, IEEE J. Sel. Areas Commun. 38 (5) (2020) 942–954.
[10] X. Hu, X. Song, G. Cheng, J. Gong, L. Yang, H. Chen, Z. Liang, Towards efficient co-audit of privacy-preserving data on consortium blockchain via group key agreement, in: The 17th International Conference on Mobility, Sensing and Networking, IEEE, 2021, pp. 1–8.
[11] M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds, IEEE J. Sel. Areas Commun. 38 (2020) 1229–1241.
[12] A. Zhang, X. Lin, Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain, J. Med. Syst. 42 (2018) 1–18.
[13] M. Yin, J. Gao, X. Guo, M. Sun, Z. Liu, J. Zhang, J. Wang, CROSS: Supervised sharing of private data over blockchains, in: International Conference on Security and Privacy in New Computing Environments, Springer, 2019, pp. 73–86.
[14] T.H. Yuen, Pachain: Private, authenticated & auditable consortium blockchain and its implementation, Future Gener. Comput. Syst. 112 (2020) 913–929.
[15] S. Malik, V. Dedeoglu, S.S. Kanhere, R. Jurdak, Privchain: Provenance and privacy preservation in blockchain enabled supply chains, 2021, CoRR, abs/2104.13964.
[16] S. Xu, X. Cai, Y. Zhao, Z. Ren, L. Du, Q. Wang, J. Zhou, Zkrpchain: Towards multi-party privacy-preserving data auditing for consortium blockchains based on zero-knowledge range proofs, Future Gener. Comput. Syst. 128 (2022) 490–504.
[17] A. Guerrero-Sánchez, E.A. Rivas-Araiza, J.L. Gonzalez-Cordoba, M. Toledano-Ayala, A. Takács, Blockchain mechanism and symmetric encryption in a wireless sensor network, Sensors (Basel, Switz.) 20 (2020) 2798.

[18] H.T.T. Truong, M. Almeida, G. Karame, C. Soriente, Towards secure and decentralized sharing of IoT data, in: 2019 IEEE International Conference on Blockchain (Blockchain), IEEE, 2019, pp. 176–183.
[19] L. Xu, T. Bao, L. Zhu, Blockchain empowered differentially private and auditable data publishing in industrial IoT, IEEE Trans. Ind. Inf. 17 (2021) 7659–7668.
[20] Z. Ma, L. Wang, W. Zhao, Blockchain-driven trusted data sharing with privacy protection in IoT sensor network, IEEE Sens. J. 21 (2021) 25472–25479.
[21] D. Han, Y. Zhu, D. Li, W. Liang, A. Souri, K.-C. Li, A blockchain-based auditable access control system for private data in service-centric IoT environments, IEEE Trans. Ind. Inf. 18 (2022) 3530–3540.
[22] W. Diffie, M. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory 22 (6) (1976) 644–654.
[23] M. Burmester, Y. Desmedt, A secure and efficient conference key distribution system, in: Workshop on the Theory and Application of of Cryptographic Techniques, Springer, 1994, pp. 275–286.
[24] M. Just, S. Vaudenay, Authenticated multi-party key agreement, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 1996, pp. 36–49.
[25] A. Shamir, Identity-based cryptosystems and signature schemes, in: Workshop on the Theory and Application of Cryptographic Techniques, Springer, 1984, pp. 47–53.
[26] S.S. Al-Riyami, K.G. Paterson, Certificateless public key cryptography, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2003, pp. 452–473.
[27] S. Heo, Z. Kim, K. Kim, Certificateless authenticated group key agreement protocol for dynamic groups, in: IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference, IEEE, 2007, pp. 464–468.
[28] J. Teng, C. Wu, A provable authenticated certificateless group key agreement with constant rounds, J. Commun. Netw. 14 (1) (2012) 104–110.
[29] B. Semal, K. Markantonakis, R.N. Akram, A certificateless group authenticated key agreement protocol for secure communication in untrusted UAV networks, in: 2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), IEEE, 2018, pp. 1–8.
[30] Q. Wu, Y. Mu, W. Susilo, B. Qin, J. Domingo-Ferrer, Asymmetric group key agreement, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2009, pp. 153–170.
[31] L. Zhang, Q. Wu, B. Qin, J. Domingo-Ferrer, Identity-based authenticated asymmetric group key agreement protocol, in: International Computing and Combinatorics Conference, Springer, 2010, pp. 510–519.
[32] X. Zhao, F. Zhang, H. Tian, Dynamic asymmetric group key agreement for ad hoc networks, Ad Hoc Netw. 9 (5) (2011) 928–939.
[33] R. Chen, J. Chen, Y. Zhang, L. Dang, Certificateless asymmetric group key agreement, J. Cryptol. Res. 3 (4) (2016) 382–398.
[34] Q. Zhang, Y. Li, R. Wang, J. Li, Y. Gan, Y. Zhang, X. Yu, Blockchain-based asymmetric group key agreement protocol for internet of vehicles, Comput. Electr. Eng. 86 (2020) 106713.
[35] Z. Xu, F. Li, H. Deng, M. Tan, J. Zhang, J. Xu, A blockchain-based authentication and dynamic group key agreement protocol, Sensors 20 (17) (2020) 4835.
[36] X. Li, Y. Wang, P. Vijayakumar, D. He, N. Kumar, J. Ma, Blockchain-based mutual-healing group key distribution scheme in unmanned aerial vehicles ad-hoc network, IEEE Trans. Veh. Technol. 68 (11) (2019) 11309–11322.

Xiaoyan Hu received the Ph.D. degree in computer architecture from Southeast University, Nanjing, China, in 2015. She visited the NetSec Laboratory, Colorado State University from 2010 to 2012. She is currently an Associate Professor with the School of Cyber Science and Engineering, Southeast University. Her research interests include blockchain technology, network traffic analysis, Internet of Things, network security, and future network architecture

Xiaoyi Song received the B.S. degree in computer science and technology from Zhongnan University of Economics and Law, Wuhan, China, in 2019. She is pursuing the M.S. degree in network cybersecurity with the School of Cyber Science and Engineering, Southeast University, Nanjing, China. Her research interest is blockchain technology.

Guang Cheng received the Ph.D. degree in computer science from Southeast University, Nanjing, China, in 2003. From 2006 to 2007, he held a post-doctoral position with the School of Electrical and Computer Engineering, Georgia Institute of Technology. He is currently a Full Professor with the School of Cyber Science and Engineering, Southeast University. His research interests include blockchain technology, network security, network measurement, and traffic sampling in computer network.

Hua Wu received the Ph.D. degree in computer application technology from Southeast University, Nanjing, China, in 2010. She is currently an Associate Professor with the School of Cyber Science and Engineering, Southeast University. Her research interests include blockchain technology, computer network security and encrypted traffic analysis.

Jian Gong received the B.S. degree in computer software from Nanjing University in 1982 and the Ph.D. degree in computer science and technology from Southeast University, Nanjing, China, in 1995. He is currently a Full Professor with the School of Cyber Science and Engineering, Southeast University. His research interests are network architecture, network intrusion detection, and network management.