symmetry

MDPI

Article

SDSM: Secure Data Sharing for Multilevel Partnerships in IoT Based Supply Chain

Chuntang Yu [1,*], Yongzhao Zhan [1] and Muhammad Sohail [2]

[1] School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang 212013, China
[2] Department of Computer Software Engineering, MCS, National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
* Correspondence: yuct@ujs.edu.cn

Abstract: Symmetric encryption algorithms enable rapid encryption of data in IoT based supply chains, which helps to alleviate the concerns of supply chain participants about privacy disclosure when sharing data. However, in supply chain management where multilevel partnerships exist universally, a pure symmetric encryption scheme cannot provide efficient data sharing and fine-grained access control. To overcome these problems, this paper proposes a secure data sharing scheme (SDSM) for IoT based supply chains by combining blockchain and ciphertext-based attribute cryptography. This scheme supports the enforcement of fine-grained access control for different levels of partnerships. In addition, to identify partnerships, we propose a metric based on the historical transaction facts on the blockchain, where the level of partnerships among participants is automatically calculated by smart contracts. Finally, we introduce personalized attributes of participants in the ciphertext-based attribute encryption algorithm to support the construction of access policies that include partnerships, allowing for more fine-grained access control. Security analyses and simulation experiments show that our proposed scheme is secure, effective, and practical.

Keywords: access control; supply chain; blockchain; attribute-based encryption

Citation: Yu, C.; Zhan, Y.; Sohail, M. SDSM: Secure Data Sharing for Multilevel Partnerships in IoT Based Supply Chain. *Symmetry* **2022**, *14*, 2656. https://doi.org/10.3390/sym14122656

Academic Editors: Lorentz Jäntschi, Liangmin Wang, Keyang Cheng and Haiqin Wu

Received: 22 October 2022
Accepted: 6 December 2022
Published: 15 December 2022

1. Introduction

With the integration of IoT technology, supply chain management can not only improve process visibility and product traceability, but also generate large amounts of data that can be shared among partners to support critical decisions in business operations [1,2]. For privacy and data security reasons, supply chain participants are prudent in picking the recipient of shared data. In supply chains, the closeness of the partnership is often the most worthwhile criterion to consider. To reduce costs, participants often host data on third-party storage platforms (e.g., cloud storage) to share them [3,4]. Before that, participants normally employ symmetric encryption algorithms or asymmetric encryption algorithms to encrypt the data to protect their business privacy. If symmetric encryption algorithms are chosen, the keys also need to be securely distributed to each recipient of the data. This typical data sharing model suffers from several problems that cannot be ignored, such as the inability of data recipients to ensure the authenticity of the data and the low efficiency of data sharing. Therefore, how to share data hosted on third-party platforms to business partners in the supply chain in a trusted, secure, and flexible manner is a hot issue of concern for both academia and industry [5–7].

The emergence of blockchain technology has brought a new opportunity for supply chain data sharing. Blockchain technology provides a better solution for those transaction scenarios that require the participation of multiple parties and are difficult to establish trust because of its advantages such as distributed storage, evidence of tampering, and multi-party consensus verification of transactions. With the help of blockchain technology, supply chain management not only addresses the issue of mutual trust in data by recording

transaction data authentically in the block, but also links all participants together and solves the problem of information silos [8–10]. All transaction records stored on the block are anchored to IoT data stored in third-party platforms through data pointers (e.g., storage address, key, signature), providing participants with a trusted data source for sharing. These trusted data require a carefully designed access policy from supply chain participants to share to the most appropriate partners. This means that participants must adopt certain methods to measure the level of partnerships [11,12]. Although blockchain brings many advantages to supply chain management, it does not solve the problem of metrics for multilevel partnerships in supply chains. It would not only significantly increase the workload of participants by having them manage a hierarchical list of partners themselves and develop access policies for shared data based on this list, but it would also be inflexible.

In order to securely and flexibly share data to one's partners, ciphertext-based attribute encryption (CP-ABE) [13,14] has gained popularity in recent years in areas such as supply chains and medical records [15–17]. The central idea of the ciphertext-based attribute encryption is to represent a user's identity as a set of attributes and then encrypt access control policies based on these attributes into the ciphertext. The encrypted data can only be decrypted if the set of identity attributes (e.g., company nationality, vendor category, etc.) of the data requester matches the access control policy. The ciphertext-based attribute encryption scheme enables a one-to-many encryption model that supports fine-grained access control of data while ensuring data security. Although the ciphertext-based attribute encryption provides sufficient flexibility, it still does not meet the needs of supply chain participants to express "partnership" when sharing data. This is because the ciphertext-based attribute encryption typically constructs access policies based on the system's predefined set of attributes and is not good at describing the personalized attributes associated with data owners. For example, as shown in Figure 1, a product manufacturer Alice can describe the access policy of its data as "Supplier", "USA", "Partner Level $\geq$ 3", where "Partner Level $\geq$ 3" is a personalized attribute closely related to Alice. This is not a situation that would be appreciated by the ciphertext-based attribute encryption.

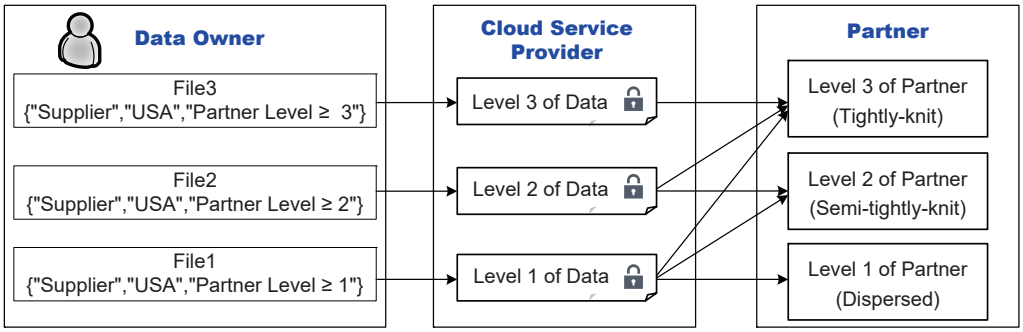


Figure 1. An example of secure data sharing based on partnership levels.

In this paper, we incorporate blockchain technology and attribute encryption technology to propose a novel data sharing scheme for IoT based supply chain management. The scheme supports each participant in the supply chain to accomplish secure and efficient data sharing according to the level of their partnership. The main contributions of this paper are summarized as follows.

(1) We propose a secure data sharing scheme (SDSM) for multilevel partnerships in IoT based supply chains. The scheme allows supply chain participants to devise access control policies based on the level of partnerships. To the best of our knowledge, this is the first secure data sharing scheme proposed for partnerships in the supply chain.

(2) We provide a definition of multilevel partnerships based on the historical transaction records between participants and propose a method for partnership measurement.

(3) We extend the ciphertext-based attribute encryption algorithm for expressing partnerships by introducing personalized attributes of participants (or PA-CP-ABE, for short). Participants can design access policies containing partnerships through personalized attributes.

(4) We perform security and privacy analyses of the proposed scheme and implement it on the Hyperledger Fabric platform to evaluate its effectiveness and feasibility.

The rest of this paper is organized as follows. We review research work related to multilevel partnerships and data sharing in Section 2. We describe the system model and design goals in Section 3. In Section 4, we present in detail our proposed secure data sharing scheme. In Section 5, we analyze the security of the proposed scheme. In Section 6, we evaluate the performance of the scheme through simulation experiments. Finally, the concluding remarks are presented in Section 7.

2. Related Work

In this section, we review the application of IoT technologies and blockchain technologies in supply chain management and highlight some of the work related to secure partnership-based data sharing in supply chains.

IoT Based Supply Chain Management. The first opportunity that IoT technology brings to the supply chain is the application of radio frequency identification (RFID) technology. Utilizing RFID and wireless communication, supply chain management can construct a global network for automatic identification and real-time sharing of item information. Niederman et al. [18] analyzed the impact of RFID technology on supply chains in terms of technical infrastructure, application logic, business processes, and management. With the addition of IoT devices such as GPS and sensors, supply chain management can not only track the location of products or individual components, but also verify and ensure the authenticity and quality of products during production and transportation. Yang et al. [19] developed a novel RFID-based system, CDTA, which is suitable for counterfeit detection, traceability, and authentication in IoT supply chains. CDTA consists of different types of on-chip sensors and in-system structures that collect the necessary information to detect multiple counterfeit IC types (recalls, clones, etc.), track and trace IoT devices, and verify the authenticity of the entire system. Further, with the aid of cloud computing, IoT is able to automate key aspects of supply chain work, such as material collection, environmental assessment, and quality inspection, greatly simplifying the process of information processing. Misra et al. [20] discussed the role of IoT and big data analytics in agriculture, supply chain modernization, and food quality assessment, and pointed out that the food supply chain industry is at the forefront of IoT applications. The IoT brings numerous obvious benefits to the supply chain, but an important prerequisite for reaping these benefits is that the data held by the various participants can be efficiently shared among partners.

Multilevel Partnerships in Supply Chains. Partnerships can drive supply chain management capabilities by integrating internal and external resources and can be classified into different levels based on criteria such as importance, strength, and closeness. Piltan et al. [21] proposed a multi-criteria decision support model to assess the factors that influence ongoing partnerships. Rezaei et al. [22] reviewed different types of partnerships and collaboration structures among networked organizations in supply chains. Participants are at each level of the partnership network and are responsible for the overall output of the network and customer satisfaction. In this structure, supply chain partners seek to gain advantages over what each would gain individually. Kim et al. [23] examined how the use of blockchain in supply chain activities affects (increases or decreases) the efficiency and growth of supply chain partnerships and thus supply chain performance outcomes. Putra et al. [24], for cross-sectoral public-private partnerships (PPPs), proposed a protocol called Putra-Ramli Secure Cyber-incident Information Sharing (PURA-SCIS) to guarantee privacy protection. While these works explored and attempted to model multilevel partnerships, they do not give how to evaluate the level of partnerships in supply chains in practice.

Secure Data Sharing. Security is particularly important when the data to be shared involves personal or business privacy. A considerable amount of research literature has proposed different approaches to securely share data. Among these approaches, ciphertext-based attribute encryption schemes have been favored by the academic community in

recent years. Qi et al.[25] proposed a scalable item-level data access control mechanism that defines and enforces access policies based on the role attributes of the participants and the RFID tag attributes of the items. To implement this mechanism, an updatable encryption scheme was designed to encapsulate the private data after attribute-based encryption. Although this encryption scheme supports item-level access control and revocation of access rights, it cannot support partnership-based hierarchical definition of access policies between upstream and downstream of the supply chain. Qi et al. [26] extended the data access control scheme in another paper to design a secure industrial data access control scheme for cloud-assisted IIoT (Industrial Internet of Things). The scheme enables participants to enforce fine-grained access control policies on their IoT data through an encryption scheme based on ciphertext policy attributes. To provide data sharing services across multiple IoT systems, Wei et al. [27] used blockchain technology to build a multicentric data management framework and designed an attribute encryption algorithm that can be used in multicentric scenarios. To improve the security and reliability of data sharing, Almagrabi et al. [28] adopted classification learning to classify the authorized resources regularly based on the trust level of available resources. Miao et al. [29] sought to protect the privacy of data providers in IoT by introducing peer-to-peer joint learning and blockchain technology to data sharing. Jia et al. [30] proposed a consensus-based distributed auction scheme for achieving privacy protection and resisting collusion attacks of shared data. Below are a few data sharing schemes with security or sensitivity levels that are most relevant to the work in this paper. Wang et al. [31] proposed an efficient hierarchical attribute encryption scheme for cloud computing files (FH-CP-ABE). This scheme integrates the hierarchical access structure into a single access structure and then uses the integrated access structure to encrypt the hierarchical files. The cryptographic components related to attributes can be shared by files. Zaghloul et al. [32] proposed a permission based multilevel organization data sharing scheme (P-MOD), which combines the permission based access structure with the attribute based encryption mechanism to handle the management and sharing of large datasets. Based on P-MOD, Zaghloul et al. further provided a hierarchical sharing scheme for medical records in [33]. In this scheme, patients are allowed to selectively share medical records based on the different levels of the staff in a hierarchical institution.

3. System Model and Problem Description

In blockchain-enabled supply chain management, the traceability of product transaction processes and the shareability of transaction data are always two very important concerns. The former can help provide the proof of product origin and the recourse for product safety incidents (e.g., fake vaccines), whereas the latter can provide benefits for decision optimization in the production process or sales execution of the involved parties. To facilitate product traceability, each product is typically assigned a code (such as an RFID code) that serves as a unique identifier for the product in offline and online transactions [18]. The participants in the supply chain associated with this product submit the transaction records to the blockchain as a non-repudiation deposit. At the same time, these transaction records link all participants together to form a traceable chain. The IoT data associated with these transaction records (e.g., location and environmental information during transportation) also hold significant business value. When it comes to sharing these data, the participants are in a dilemma. On the one hand, they expect to achieve productivity and sales efficiency by sharing data among collaborators. On the other hand, they want to ensure that the privacy of their business is not compromised to unrelated users, especially their business rivals.

Encrypting the data before sharing them seems to be a more straightforward idea, and then the participants in supply chains host the encrypted data on a third-party cloud platform (for cost reasons) [26]. We refer to the tuple $\langle CT_d^{addr}, K_{sym} \rangle$ as a data pointer where CT_d^{addr} is the address where the encrypted data are stored on the cloud platform, and K_{sym} is the key used for encryption. What needs to be considered later is how to share these data

pointers effectively to those partners that the data provider expects. Our first thought is to submit these data pointers to the blockchain in the form of transactions (named point transactions). Due to the openness of blockchain data storage, any legitimate user of the blockchain is able to extract the transaction data in the block. To solve this problem, we encrypt the data pointers once more before committing them to the blockchain. This time, the ciphertext-based attribute encryption becomes our ideal encryption scheme because it has the advantage of being encrypted once and shared by many.

In a point transaction, we often share multiple data related to the product, such as price, location information in transportation, monitoring data of the storage environment, and so on. These shared data have different sensitivities. We store them in different files, denoted by $F_i(i = 1, 2, \cdots, n)$, where n is the highest sensitivity level of the file [32]. The participants sharing data expect that the files with higher sensitivities require higher levels of partnerships to access them. We use $\mathcal{L}_i(i = 1, 2, \cdots, n)$ to denote the partnership level. The larger the value of i, the higher the partnership level.

3.1. System Model

The system model of a secure data sharing scheme (SDSM) for multilevel partnerships is shown in Figure 2. This scheme consists of six key components.

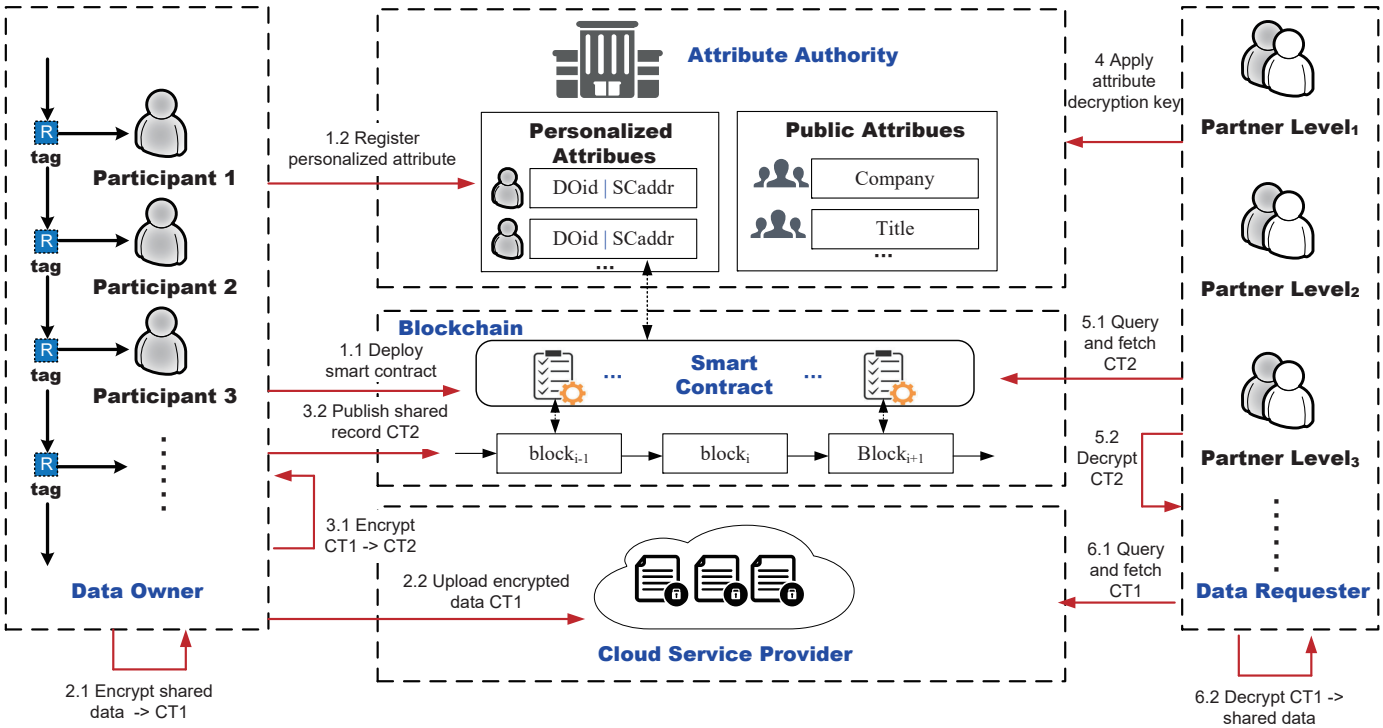


Figure 2. System model.

(1) *Data Owner (DO).* A data owner is a provider of shared data. In our supply chain management, all participants who complete product transactions in the supply chain can be considered as data owners. Before sharing data, they can request their own personalized attributes from the attribute manager and integrate these attributes into their desired access policies. We denote the set of DO as $\mathbb{DO} = \{DO_i | 1 \leq i \leq n\}$, where DO_i denotes the ith data owner. We denote the set of personalized attributes as $\mathbb{PA} = \{PA_i | 1 \leq i \leq n\}$, where PA_i is the personalized attribute of the DO_i; PA_i is a multi-valued attribute, and each value corresponds to a partnership level $\mathcal{L}_j$. For simplicity, we specify that when $PA_i = x$, it denotes the xth level partnership $\mathcal{L}_x$.

(2) *Data Requester (DR).* A data requester is a visitor to the shared data provided by data owners. In our supply chain management, all supply chain participants (including

possible regulators) can be data requesters. We denote the set of DR as $\mathbb{DR} = \{DR_i | 1 \leq i \leq n\}$, where DR_i denotes the ith data requester. Each data requester can request a decryption private key based on his own set $\mathbb{S} = \{S_1, S_2, \cdots, S_k\}$ of attributes. In addition to this, the data requester can apply the decryption private key by combining $\mathbb{S}$ with the personalized attribute PA_i of the particular data owner DO_i.

(3) *Attribute Authority (AA).* The attribute authority is the administrator of the system attributes and is responsible for verifying the attributes for all data requesters and issuing private keys for data decryption. In our scheme, the attribute authority may also need to interact with smart contracts to evalute the partnership level of the data requester DR_j with respect to the data owner DO_i.

(4) *Smart Contract (SC).* Smart contracts are used to measure partnerships between participants. In our scheme, these smart contracts are created and deployed by the system and data owners on the blockchain. In particular, the system contract provides some basic methods for measuring partnerships. The data owner's private contract completes the grading of the partnership by invoking these basic methods. These private contracts will be invoked by the attribute authority in the future.

(5) *Blockchain (BC).* Blockchain provides a storage space for two kinds of transaction data and an execution environment for smart contracts. These two types of transactions include product ownership-related transactions and data sharing-related point transactions. By the point transaction, we mean the posting of data encryption pointers to the blockchain.

(6) *Cloud Service Provider (CSP).* The CSP provides the initial storage space for shared data and is ready to respond to data access requests from data owners and data requesters.

3.2. Design Goals

Our goal is to design an access control scheme for blockchain-enabled supply chains that supports multilevels partnerships and personalized attributes. Different data owners are able to assign access policies to their records so that only specific authorized partners in the supply chain can access them. We have identified three design requirements that a supply chain management system should support:

- *Data privacy:* The most basic design requirement is to prevent unauthorized participants from viewing any important information about the shared data submitted by the data owner.
- *Fine-grained access control:* Each data owner involved in a product transaction is able to specify an access policy for his shared data related to product transactions. The policy should be granular enough to help the data owner define an access policy based on the system's predefined set of attributes and his own personalized attributes to accurately describe his authorized partners.
- *Resistant to collusion attacks:* Two or more data requesters with different attributes cannot combine their attributes to access and decrypt data for which they are not authorized by a data owner.

3.3. Security Model

In our SDSM scheme, the ciphertext-based attribute encryption algorithm is extended to support the personalized attributes of data owners (PA-CP-ABE). The extended PA-CP-ABE needs to satisfy the following security model [13].

Suppose $\mathcal{B}$ acts as the challenger and $\mathcal{A}$ acts as the adversary in the game. We present the PA-CP-ABE game as follows:

Init. Challenger $\mathcal{B}$ takes in a q-parallel BDHE challenge. Adversary $\mathcal{A}$ gives the challenge access structure $(\boldsymbol{M}^*, \rho^*)$ to the algorithm.

Setup. Challenger $\mathcal{B}$ runs the Setup algorithm and gives the public parameters, PK to adversary $\mathcal{A}$.

Query 1. Adversary $\mathcal{A}$ sends multiple sets of attributes $S_1, S_2, \cdots, S_{q_1}$ to challenger $\mathcal{B}$, and these sets of attributes cannot satisfy the access policy $(\boldsymbol{M}^*, \rho^*)$. Challenger $\mathcal{B}$ runs the

private key generation algorithm KeyGen to generate the corresponding attribute private key and sends it to adversary $\mathcal{A}$.

Challenge. Adversary $\mathcal{A}$ submits two plaintext messages m_0 and m_1 of equal length. Challenger $\mathcal{B}$ randomly selects $c \in \{0,1\}$ and encrypts m_c under the attribute access policy $(\boldsymbol{M}^*, \rho^*)$. The generated ciphertext CT^* is sent to adversary $\mathcal{A}$.

Query 2. Same as **Query 1.** Adversary $\mathcal{A}$ continues to submit key queries for a series of attribute sets $S_{q_1+1}, S_{q_1+2}, \cdots, S_q$, again requiring that none of these attribute sets satisfy the access structure $(\boldsymbol{M}^*, \rho^*)$.

Guess. Adversary $\mathcal{A}$ outputs a guess c' of c and defines $|\Pr[c = c'] - \frac{1}{2}|$ as the advantage of adversary $\mathcal{A}$ in this game.

Definition 1 (Selective Access Structure Secure)**.** *A PA-CP-ABE algorithm is Selective Access Structure Secure if no polynomial-time adversary $\mathcal{A}$ wins the above game with the advantage $|\Pr[c' = c] - \frac{1}{2}| > \varepsilon(\cdot)$, where $\varepsilon(\cdot)$ is non-negligible function.*

4. The Proposed SDSM Scheme

4.1. Definition of Multilevel Partnership

The criteria for classifying partnership levels in the supply chain were discussed in great detail in the literature [21,22]. From these works we can conclude that the number of transactions completed between supply chain participants in a given period of time is a core indicator for determining the partnership level.

In our scheme, the blockchain involves two types of transaction records: product transaction records and point transaction records. Our definition of the partnership is based on the history of blockchain product transactions. A product transaction record can be represented as a tuple $\langle p_{id}, send_{id}, recv_{id}, sign, timestamp \rangle$, where p_{id} is the product identifier, $send_{id}$ is the id of the transaction initiator, $recv_{id}$ is the id of the transaction recipient, $sign$ is the signature of the transaction initiator on the transaction record, and $timestamp$ is the time when the transaction is recorded on the blockchain. It should be emphasized that the initiator of a product transaction is also the data owner of a point transaction, and the recipient of a product transaction is also the data requester of a point transaction. In this paper, we assume that each participant appears at most once each as an initiator or a recipient in the chain of transactions for a product. The chain of transactions for the product p_{id} is referred to as $\mathbb{PC}_{p_{id}}$. The index of transaction record Tx in the transaction chain is denoted as $index(Tx)$.

Definition 2 (Transaction Distance)**.** *For the two transaction records Tx_a and Tx_b that successively appear in the transaction chain $\mathbb{PC}_{p_{id}}$ of the product p_{id}, the **Transaction Distance** between the initiator $send_{id}$ of Tx_a and the recipient $recv_{id}$ of Tx_b is defined as **TxD**$(send_{id}, recv_{id}, p_{id})$ = index (Tx_b) − index (Tx_a), as shown in Figure 3. If $send_{id}$ and $recv_{id}$ do not appear in the same transaction chain, we specify **TxD**$(send_{id}, recv_{id}, p_{id}) = \infty$.*

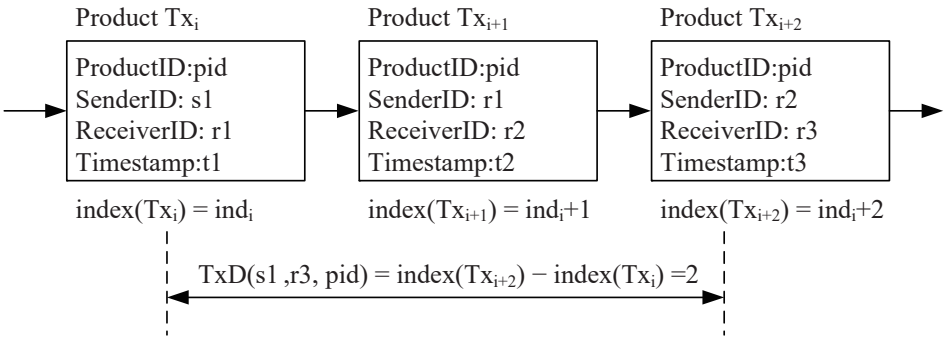


Figure 3. Transaction distance (TxD).

When we are talking about the sharing of point transaction records, they can also be written as $\mathbf{TxD}(DO_{id}, DR_{id}, p_{id})$. Here, DO_{id} and DR_{id} are the identifiers of the data owner and data requester, respectively.

Definition 3 (Multilevel Partnership)**.** *For the data owner DO and the data requester DR, if they appear in the transaction chain of a product and meet specific transaction distance constraints τ in a certain period of time in the past, we say that the DO and DR have an effective partner connection qpc. The total number $sum(qpc)$ of such effective partner connections is recorded as η. Here, τ is an integer interval in the form of [1,3], representing the lower and upper bounds of the transaction distance. The data owner DO divides η into multiple levels $\eta_i (i = 0, 1, \cdots, l)$ and each level η_i corresponds to a level $\mathcal{L}_i$ of partnerships.* ***Multilevel Partnership*** *is defined as $\mathcal{L} = \{\mathcal{L}_i | sum(qpc) \geq \eta_i, i = 0, 1, \cdots, l\}$, where l is the highest level.*

The difference between our scheme and the other two related works [31,32] is in the identification of multilevel relationships. Our multilevel partnerships are related to the data owners, whereas the ones in [31,32] are not. The literature [24] talked about public–private partnerships across organizational sectors, but did not give a specific division method.

4.2. The SDSM Construction

As shown in Figure 2, the participants of the supply chain are both data owners and data requesters. In a practical scenario, data requesters possibly also include regulators and consumers. For data storage, we employ a cloud service provider, which is an honest and curious entity. The cloud service provider and the blockchain platform jointly maintain the storage of transaction data. The shared original data are stored in the cloud platform after symmetric encryption, and the blockchain stores the data pointers. Before the data pointer is submitted to the blockchain platform (a point transaction), the data owner performs attribute encryption on the data pointer. The execution process of the whole scheme can be divided into the following major events.

(1) *Register Personalized Attribute.* In this paper, a personalized attribute refers to an attribute of a data requester that is associated with a particular data owner. If a supply chain participant requires a personalized attribute to provide more precise privacy protection for his transaction records when sharing transaction data, then he is allowed to request his personalized attributes from the attribute authority. For these, he first creates a smart contract $DOSC$ following the approach in Section 4.3 and deploys it on the blockchain. This smart contract is used to calculate the partnership level $\mathcal{L}$ of the data requester. Then, he sends the message $(DO_{id}, DOSC_{addr}, l)$ to the attribute authority, where DO_{id} is the identifier of the DO, $DOSC_{addr}$ is the address of the smart contract on the blockchain, and l is the highest partnership level. The attribute authority verifies the DO's identity and creates a multi-valued attribute for him.

(2) *Encrypt Shared Data.* The data owner splits the batch of data to be shared into multiple files according to the sensitivity level from low to high, $F_{s_1}, F_{s_2}, \ldots, F_{s_m} (0 < s_1 < s_2 < \cdots < s_m \leq l)$. For each file $F_{s_i} (i = 0, 1, \cdots m)$, the data owner encrypts it using a symmetric encryption algorithm as

$$CT_{F_{s_i}} = Enc_{sym}(F_{s_i}, K_{sym_i}), \tag{1}$$

where Enc_{sym} is a symmetric encryption algorithm such as Advanced Encryption Algorithm (AES), and K_{sym_i} is the key chosen by the data owner to perform symmetric encryption of the shared data. Inspired by [32], the data owner randomly chooses the key K_{sym_m}, and the encryption keys for other files of lower sensitivity are derived according to the following rules

$$K_{sym_i} = H(K_{sym_{i+1}}), \tag{2}$$

where H is a hash function. The advantage of this is that high-level partners can derive decryption keys for low-sensitivity level files.

Next, the data owner uploads the encrypted files $CT_{F_{s_i}}$ to the cloud service provider for storage, and the cloud service provider returns the storage address $CT^{addr}_{F_{s_i}}$ of the data to the publisher. In this way, the data owner holds a collection of addresses and key pairs $\{\langle CT^{addr}_{F_{s_i}}, K_{sym_i}\rangle | 0 < i \leq m\}$. We can also refer to $\langle CT^{addr}_{F_{s_i}}, K_{sym_i}\rangle$ as the data pointer of the file.

(3) *Publish Shared Record.* For each K_{sym_i}, the data owner engages in encryption using an attribute encryption algorithm. The data owner designs different attribute encryption policies for different levels of partners based on the system's predefined attributes and his personalized attributes to protect his data from unauthorized access. As a limitation, in this paper, we assume that the visitors of a shared file in the same publishing have the same attributes, except for different partnership levels. This assumption is consistent with the characteristics of file sharing in the supply chain in practice. The encryption process is

$$CT_{K_i} = Enc_{abe}(K_{sym_i}, \rho_i), \tag{3}$$

where Enc_{abe} is the encryption algorithm we defined in the scheme description, and ρ_i is the access policy defined by the data owner. Finally, the data owner initiates a point transaction to the blockchain, storing the transaction record permanently on the blockchain. The message of the point transaction is $(\{\langle CT^{addr}_{F_{s_i}}, CT_{K_i}\rangle | 0 < i \leq m\}, sign_{DO}, timestamp)$, where $sign_{DO}(\{\langle CT_{K_i}, CT^{addr}_{F_{s_i}}\rangle | 0 < i \leq m\}, sign_{DO}, timestamp)$, where $sign_{DO}$ is DO's signature on the message. We can also refer to $\langle CT^{addr}_{F_{s_i}}, CT_{K_i}\rangle$ as the encrypted data pointer of the file.

(4) *Apply Attribute Decryption Key.* Once a data requester perceives that he can benefit from the data shared by some participant, in order to obtain the original record, he is required to apply a decryption key from the attribute authority. This decryption key is based on the attributes of the data requester, and, in our scenario, may also require the existence of some business partnership between the requester and the sharer of the data. The data requester provides a proof of his attributes to the attribute authority, as well as some personalized attributes that are of interest to him. The proof of personalized attributes does not need to be provided by the requester; it will be automatically verified by the attribute authority by invoking a smart contract. After completing the verification of the requester's attributes, the attribute authority generates the decrypted private key K_{abe}.

(5) *Fetch Symmetric Key.* The data requester downloads the point transaction record from the blockchain and reads the attribute-encrypted portion CT_{K_i}. With the decryption private key K_{abe} obtained from the attribute authority, it attempts to decrypt CT_{K_i}. If the decryption key K_{abe} satisfies the access policy in CT_{K_i}, the decryption operation will finish successfully as

$$K_{sym_i} = Dec_{abe}(CT_{K_i}, K_{abe}), \tag{4}$$

where Dec_{abe} is the attribute decryption function.

(6) *Extract Shared Data.* According to the data pointer address $CT^{addr}_{F_{s_i}}$, the data requester is able to download the data $CT_{F_{s_i}}$ from the cloud service provider. Potentially, our paper assumes that the data requester already has the data access rights granted by the cloud service provider. In the next step, the data requester achieves the decryption of the data aided by the symmetric key such that

$$F_{s_i} = Dec_{sym}(CT_{F_{s_i}}, K_{sym_i}), \tag{5}$$

where Dec_{sym} is the symmetric decryption function. If $i > 1$, the data requester calculates the lower-level file encryption key according to Equation (2) and decrypts it using Equation (5).

At this point, the data requester has completed the entire decryption process of the data. The supply chain participant who published the data has successfully shared his private data to this data requester.

4.3. Smart Contracts

In our proposed scheme, the blockchain is not only intended for storing the encrypted pointers to shared data, but also for computing the partnerships between data requesters and data owners. Based on the proposed concept of multilevel partnerships, we validate the partnerships by designing and deploying smart contracts. These smart contracts are divided into two types: system contracts and user contracts. System contracts are implemented to calculate the transaction distance, whereas user contracts define their own partnerships by invoking system contracts.

System Contracts. Because the calculation of the transaction distance is a standardized process, we implement it in the form of a system contract. Algorithm 1 outlines the computational function of the transaction distance. Lines 3–7 query all transactions performed after the time *startTime* based on the product p_{id}. The function ***SearchTransByPid*** will be the most time-consuming operation in this algorithm because it has to traverse all the transactions if no index exists in the historical transaction database. Lines 14–21 accomplish the function of determining the index of the location of the data owner DO_{id} and the data requester DR_{id} in the transaction chain. Finally, the transaction distance is calculated in line 26. In our paper, the transaction distance does not take into account the positivity and negativity.

Algorithm 1: ComputeTxD

Input: $DO_{id}, DR_{id}, p_{id}, startTime$/* DO_{id}: data owner id, DR_{id}: data requester id, p_{id}: product id and *startTime*: earliest time for transaction submission */
Output: an integer representing the transaction distance

```
1  Create a product transaction list tranList
2  /*Search for transactions related to pid*/
3  tranList = SearchTransByPid(p_id)
4  for tran in tranList do
5      if tran.timestamp < startTime then
6          tranList.remove(tran)
7      end
8  end
9  /*The transaction is sorted in ascending chronological order*/
10 tranList = OrderTransByTime(tranList)
11 indexDO_id ← ∞
12 indexDR_id ← ∞
13 /*Find the index of DO and DR in the transaction chain*/
14 for tran in tranList do
15     if tran.send_id == DO_id and indexDO_id == ∞ then
16         indexDO_id = index(tran)
17     end
18     if tran.recv_id == DR_id and indexDR_id == ∞ then
19         indexDR_id = index(tran)
20     end
21 end
22 /*If DO or DR does not appear in the transaction chain, ∞ is returned*/
23 if indexDO_id == ∞ or indexDR_id == ∞ then
24     return ∞
25 end
26 txDistance = Math.abs(indexDO_id − indexDR_id)
27 return txDistance
```

User Contracts. In order to integrate partnerships in the data access policy, it is necessary for the data owner to design the logic for calculating the partnership level in their

own smart contract. Algorithm 2 outlines the computational function of the partnership level. As we described in Definition 3, the partnership level depends on the number of products traded between the data owner and the data requester under certain constraints. Line 3 specifies the mapping relationship between the number of products traded and the partnership level. Line 6 uses the function ***SearchPidByUid*** to query the ids of all products in which the data owner is involved in a transaction. Similar to ***SearchTransByPid*** in Algorithm 1, this is a time-consuming operation. Lines 8–15 calculate the transaction distance between the data owner and the data requester for all products queried before and count the number of products that satisfy the transaction distance constraint range (line 12). In this process, the algorithm ComputeTxD is called. Finally, in lines 17–21, the partnership level is returned based on the counted number of products.

Algorithm 2: Evaluate Level of Partnership

Input: $DO_{id}, DR_{id}, dis_{min}, dis_{max}, startTime$ /* DO_{id}:data owner id, DR_{id}: data requester id, dis_{min}: min transaction distance, dis_{max}: max transaction distance and $startTime$: earliest time for transaction submission*/

Output: an integer representing the level of partnership

1 /*The DO classify partnership levels based on the number of traded products*/
2 /*η_i: the number of products, $\mathcal{L}_i$: the level of partnership */
3 Create a map $Map : \eta_i \rightarrow \mathcal{L}_i (i = 1, 2, \cdots, m)$
4 $productCount \leftarrow 0$
5 /*Search all product IDs of DO's participating transactions
6 $allP_{id} =$ ***SearchPidByUid***(DO_{id})
7 /*Count the number of products that satisfy the constraint*/
8 **for** p_{id} **in** $allP_{id}$ **do**
9 /*Call the method ComputeTxD in the system contract */
10 $dist = SC_{sys}.ComputeTxD(DO_{id}, DR_{id}, p_{id}, startTime)$
11 /* Verify that the transaction distance satisfies the constraint */
12 **if** $dis_{min} \leq dist$ **and** $dist \leq dis_{max}$ **then**
13 $productCount = productCount + 1$
14 **end**
15 **end**
16 /*Determining the level of partnership*/
17 **for** $i = m$ **in** 1 **do**
18 **if** $productCount \geq \eta_i$ **then**
19 return $Map[\eta_i]$
20 **end**
21 **end**
22 return 0

4.4. PA-CP-ABE Algorithm Description

Our extended PA-CP-ABE algorithm is based on the construction presented in [13] and mainly consists of five parts: Setup, AttrDef, KeyGen, Encrypt, and Decrypt. The details are as follows.

(1) Setup$(1^\lambda) \rightarrow$ (PK, MSK). The Setup algorithm outputs the system public and private key pair (PK, MSK) by inputting the security parameter λ. There is a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, where $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups with prime order p; g is a generator of $\mathbb{G}$. This algorithm chooses a cryptographic hash function $H : \{0,1\}^* \rightarrow \mathbb{G}$, which maps a binary string describing an attribute to a hash value in the group $\mathbb{G}$. In addition, it chooses random exponents $\mu, \zeta \in \mathbb{Z}_p$, and outputs system key pair (PK,MSK) as

$$\text{PK} = (g, e(g,g)^\mu, g^\zeta), \text{MSK} = g^\mu.$$

(2) AttrDef(δ_{DO_i}, Exp) $\rightarrow PA_{DO_i}$. The AttrDef algorithm outputs the personalized attribute PA_{DO_i} for DO_i by inputting the identifier δ_{DO_i} of the data owner and the partnership conditional expression Exp; PA_{DO_i} is abbreviated to PA where there is no confusion.

(3) KeyGen(MSK, δ_{DR_j}, PA_{DO_i}, S) $\rightarrow$ SK. The attribute private key generation algorithm inputs the system private key MSK, the identifier δ_{DR_j} of the data requester DR_j, the personalized attribute PA_{DO_i}, the attribute set S of the data requester DR_j, and outputs the attribute private key SK for the data requester DR_j. Here, the inputs δ_{DR_j} and PA_{DO_i} are for verifying that the data requester satisfies the data owner's personalized attribute. This verification process is performed by the attribute authority invoking the smart contract. The algorithm chooses a random $r \in \mathbb{Z}_p$ and creates the private key as

$$\mathrm{SK} = (K = g^{\mu} g^{\zeta r}, P = g^{r}, \{K_x = H(x)^{r}\}_{x \in S \cup PA}).$$

(4) Encrypt(PK, m, $(\boldsymbol{M}, \rho)$) $\rightarrow$ CT. The encryption algorithm takes as input the public parameters PK, a message m to encrypt, and an LSSS access structure $(\boldsymbol{M}, \rho)$.

In the encryption process, $\boldsymbol{M}$ is the access matrix of $l \times n$, $\boldsymbol{M}_i$ denotes the ith row in the matrix, $i \in [1, 2, \ldots, l]$. The function ρ associates rows to attributes. The algorithm selects a random vector $\nu = (s, t_2, \ldots, t_n) \in \mathbb{Z}_p^n$ to generate ciphertext information (C, C'); s is the secret of the encryption exponent, and $t_2, \ldots, t_n$ is a series of random values. The algorithm selects random $r_1, r_2, \ldots, r_l \in \mathbb{Z}_p$ and adds additional information $(C_i, D_i)_{i \in [1,l]}$ to the ciphertext. Finally, the ciphertext CT is published as

$$\mathrm{CT} = (C = me(g,g)^{\mu s}, C' = g^{s}, (C_i = g^{\zeta \lambda_i} H(\rho(i))^{-r_i}, D_i = g^{r_i})_{i \in [1,l]}).$$

(5) Decrypt(CT,SK)$\rightarrow m$. The decryption algorithm inputs a ciphertext CT about the access policy $(\boldsymbol{M}, \rho)$, and a private key SK about the attribute set $S' = S \cup PA$ of the data requester. If the attribute set S' of the data requester satisfies the access policy $(\boldsymbol{M}, \rho)$, the plaintext m is the output, otherwise, the decryption fails.

Let the set of attribute index be $I = \{i : \rho(i) \in S'\}$ and the target vector be $(1, 0, \ldots, 0)$. If the attributes of the data requester fully meet access matrix $(\boldsymbol{M})_{i \in I}$, it is able to find a set of vectors $\{\boldsymbol{w}_i\}_{i \in I}$ such that $\sum_{i \in I} \boldsymbol{w}_i \boldsymbol{M}_i = (1, 0, \ldots 0)$, then we have $\sum_{i \in I} \boldsymbol{w}_i \lambda_i = s$. The decryption algorithm computes

$$\frac{e(C', K)}{\prod_{i \in I} (e(C_i, P) e(D_i, K_{\rho(i)}))^{w_i}} = \frac{e(g,g)^{\mu s} e(g,g)^{s \zeta r}}{\prod_{i \in I} e(g,g)^{\zeta r \lambda_i w_i}} = \frac{e(g,g)^{\mu s} e(g,g)^{s \zeta r}}{e(g,g)^{\zeta r \sum_{i \in I} w_i \lambda_i}} = e(g,g)^{\mu s}.$$

Finally, we get the $m = \frac{C}{e(g,g)^{\mu s}}$.

5. Security Analysis

5.1. Security Analysis of the PA-CP-ABE Algorithm

We formally define the desired security property based on an interactive game consisting of an adversary and a challenger. The adversary interacts with the challenger to attack our PA-CP-ABE algorithm. We then prove that our PA-CP-ABE satisfies the security property based on the paradigms in [13].

Theorem 1. *If decisional q-parallel BDHE assumption holds, then no polynomial-time adversary can choose the challenge access structure* $(\boldsymbol{M}^*, \rho^*)$ *to break the PA-CP-ABE algorithm.*

Proof. If there exists a polynomial-time adversary $\mathcal{A}$ choosing a challenge access structure $(\boldsymbol{M}^*, \rho^*)$, wins the game by a non-negligible advantage ε, then there exists a simulator $\mathcal{B}$ that solves the decisional q-parallel BDHE assumption by a non-negligible advantage ε.

(1) *Init.* $\mathcal{B}$ takes in a q-parallel BDHE challenge $\boldsymbol{y}, T$; $\mathcal{A}$ gives the algorithm the challenge access structure $(\boldsymbol{M}^*, \rho^*)$, where M^* has n^* columns, and $n^* \leq q$.

(2) *Setup.* $\mathcal{B}$ randomly selects a number $\mu' \in \mathbb{Z}_p$, and lets $\mu = \mu' + \zeta^{q+1}$ such that $e(g,g)^{\mu} = e(g,g)^{\mu'+\zeta^{q+1}} = e\left(g^{\zeta}, g\right)^{\zeta^q} e(g,g)^{\mu'}$. Next, $\mathcal{B}$ chooses a random oracle H and builds a table. When H is called, the result is returned directly if $H(x)$ already exists in the table, and if $H(x)$ does not exist in the table, $z_x \in \mathbb{Z}_p$ is chosen randomly. Let X be the index set $X = \{i : \rho^*(i) = x\}$. If x is an element in the set of system attributes, $H(x) = g^{z_x} \prod_{i \in X} (g^{\frac{\zeta M^*_{i,1}}{b_i}} . g^{\frac{\zeta^2 M^*_{i,2}}{b_i}} \cdots g^{\frac{\zeta^{n^*} M^*_{i,n^*}}{b_i}})$.

(3) *Query 1.* $\mathcal{A}$ queries the private key by submitting a set $S' = \{PA_{DO_i}, S_1, S_2, \cdots, S_n\}$, where PA_{DO_i} is the personalized attribute of a data owner and S' does not satisfy access policy $(\boldsymbol{M}^*, \rho^*)$; $\mathcal{B}$ runs the private key generation algorithm KeyGen to generate the corresponding private key SK = (K, P, K_x) to the adversary. The following is the construction process of the private key. According to the LSSS, $\mathcal{B}$ can find a vector $\boldsymbol{w} = (w_1, w_2, \cdots, w_n) \in \mathbb{Z}_p^{n^*}$ such that $\boldsymbol{w}\boldsymbol{M}^*_i = 0$, where $w_1 = -1$; $\mathcal{B}$ defines implicitly $t = r + w_1\zeta^q + w_2\zeta^{q-1} + \cdots + w^*_n\zeta^{q-n^*+1}, r \in \mathbb{Z}_p$, then it computes $P = g^r \prod_{i=1,\cdots,n^*} \left(g^{\zeta^{q+1-i}}\right)^{w_i} = g^t$. To eliminate item $g^{\zeta^{q+1}}$, $\mathcal{B}$ uses $\mu = \mu' + \zeta^{q+1}$ to define K as

$$K = g^{\mu} g^{\zeta t} = g^{\mu' + \zeta^{q+1}} g^{\zeta t} = g^{\mu'} g^{\zeta r} \prod_{i=2,\cdots,n^*} \left(g^{\zeta^{q+2-i}}\right)^{w_i}.$$

The attribute set S' submitted by the data requester DR consists of two parts: the system's predefined attribute set $S = \{S_1, S_2, \cdots, S_n\}$ and the data owner's personalized attribute set $PA = \{PA_1, PA_2, \cdots, PA_n\}$, i.e., $S' = S \cup PA$; $\mathcal{B}$ can compute K as

$$K_x = H(x)^t = P^{z_x} \prod_{i \in X} \prod_{j=1,\ldots,n^*} \left(g^{\frac{\zeta^j}{b_i} r} \prod_{\substack{m=1,\ldots,n^* \\ m \neq j}} \left(g^{\zeta^{q+1+j-m}}\right)^{w_m} \right)^{\boldsymbol{M}^*_{ij}}, x \in S$$

$$K_x = H(x)^t = g^{z_x t} = P^{z_x}, \quad x \in PA.$$

(4) *Challenge.* $\mathcal{A}$ sends two messages m_0, m_1 of equal length to $\mathcal{B}$; $\mathcal{B}$ randomly picks $c \in \{0,1\}$, then it uses the access structure $(\boldsymbol{M}, \rho)$ to encrypt m_c such that $C = m_c \mathrm{T} e(g,g)^{\mu s}$ and $C' = g^s$. Next, $\mathcal{B}$ chooses randomly $y'_2, y'_3, \cdots, y'_{n^*}$ and shares the secret key by $\boldsymbol{v} = (s, s\zeta + y^*_2, s\zeta^2 + y^*_3, \cdots, s\zeta^{n-1} + y^*_{n^*}) \in Z_p^{n^*}$. Finally, $\mathcal{B}$ chooses randomly $r'_1, \cdots, r'_{l^*} \in Z_p$ and defines R_i as the set of $\rho^*(i) = \rho^*(m)$ $(m \neq i)$. The ciphertext is created as

$$C_i = H(\rho^*(i))^{r'_i} \left(\prod_{j=2,\cdots,n^*} \left(g^{\zeta}\right)^{M^*_{i,j} y'_j} \right) \left(g^{b_i s}\right)^{-z_{p^*(i)}} . \left(\prod_{m \in R_i} \prod_{j=1,\cdots,n^*} \left(g^{\zeta^j s \frac{b_i}{b_m}} \right)^{M^*_{m,j}} \right),$$

$$D_i = g^{-r'_i} g^{-s b_i}.$$

(5) *Query 2.* Same as Query 1.

(6) *Guess.* $\mathcal{A}$ outputs a guess c' of c, where $c' \in \{0,1\}$. Next, $\mathcal{B}$ outputs $\theta = 0$ if $c' = c$ which shows that $T = e(g,g)^{\zeta^{q+1} s}$; otherwise, $\mathcal{B}$ outputs $\theta = 1$ to show that T is a random group element in $\mathbb{G}_T$. When $c' = c$, the advantage of $\mathcal{A}$ is $\Pr[c = c' | \theta = 0] = \frac{1}{2} + \varepsilon$. When $c' \neq c$, the advantage of $\mathcal{A}$ is $\Pr[c = c' | \theta = 1] = \frac{1}{2}$. The advantage gained by adversary $\mathcal{A}$ in attacking q-parallel BDHE assumption is

$$\mathrm{Adv}_{\mathcal{A}} = \left| \frac{1}{2} \Pr[c = c' | \theta = 0] + \frac{1}{2} \Pr[c = c' | \theta = 1] - \frac{1}{2} \right| = \frac{1}{2} \varepsilon.$$

Therefore, the advantage of any polynomial-time adversary in winning the game is negligible. □

5.2. Security Analysis of Our SDSM Scheme

Among the security goals of our scheme, the focus is on data privacy and resistance to collusion attacks. Recently, Jia et al. [30] proposed a consensus-based distributed auction scheme for enhancing privacy protection and resisting collusion in data sharing. Unlike this, our scheme is mainly based on cryptographic policies to reach these security goals.

Data Privacy: In our scheme, a hybrid key encapsulation mechanism is used for encryption, i.e., we encrypt the original data using the symmetric K_{sym} and then encrypt the K_{sym} again using the attribute encryption. This is similar to the method in [33]. Although the cloud service provider is a semi-trusted entity, it must obtain the session key to decrypt the shared data stored by the data owner. However, the session key has been encrypted by our PA-CP-ABE algorithm and stored on the blockchain. Due to the open nature of the blockchain, all nodes can synchronize the entire full block. After extracting the transaction data on the block, only the private key that fully satisfies the access policy can decrypt these transaction data. Therefore, our system can protect the data privacy of supply chain participants.

Resisting Collusion Attack: In the CP-ABE scheme, if the attributes of two or more data requesters do not satisfy the access policy, the decryption of data is achieved by merging their attribute keys, and we say that the scheme is not resistant to collusion attack. We introduce personalized attributes of the data owner for the CP-ABE scheme, which may serve as an entry point for attackers. However, in our PA-CP-ABE algorithm, the attribute authority aggregates these attributes to generate the attribute private keys after verifying whether the data requester satisfies the personalized attributes. That is, these attribute private keys use the same batch of random factors. When different data requesters request their own attribute private keys, the attribute authority uses different random factors, so they will not be able to complete the decryption of the ciphertext by combining their respective private keys.

6. Performance Evaluation

Our secure data sharing scheme combines attribute cryptography and blockchain technology, so we evaluated the efficiency and reliability of our scheme from two perspectives, the first involving the core algorithms in the attribute cryptography scheme, and the second involving data publishing and smart contract execution on the blockchain.

6.1. Comparison of Encryption Schemes

For the CP-ABE scheme, we used the JPBC library for the implementation and selected several related works [14,31,32] to compare with our scheme. The ciphertext-based attribute encryption scheme (CP-ABE) proposed by Bethencourt et al. [14] served as the baseline for our comparison. Our proposed attribute encryption scheme (PA-CP-ABE) was compared with the other two most relevant schemes, namely, the File Hierarchical Attribute Encryption scheme (FH-CP-ABE) proposed by Wang et al. [31] and the Permission-based Multilevel Organisation Data Encryption scheme (P-MOD) proposed by Zaghloul et al. [32]. The FH-CP-ABE scheme integrates layered access structures into a single access structure, and then encrypts the layered files with the integrated access structure. This scheme saves the cost of storing the ciphertext and the time cost of encryption, but the access structure will become relatively complex and increase the generation time of the decryption key. The P-MOD scheme requires the existence of a clear hierarchy in the organization of the data requester. This allows the data owner to design access policies based on different hierarchies and to divide the file into different sections, each assigned a different hierarchy. Each file part is encrypted with a symmetric key and is able to derive the key of the lower-level part from the key of the higher-level part.

We completed two tests on a personal computer, a laptop with a 6-core, 12-thread CPU and 32G RAM. The two tests were: (1) the attribute encryption and decryption of the transaction data; and (2) the generation of the data requester's private key.

As shown in Figure 4a,b, the time overhead of our scheme is significantly smaller than the other three schemes. This is mainly because the other three schemes use policy trees as the access structure, whereas our scheme adopts LSSS. On the other hand, the comparison between Figure 4a,d also shows that the overhead of the CP-ABE scheme is particularly high as the attribute number increases.

The experiments are conducted from the perspective of the data requester appearing at the highest level of the hierarchy. Taking this into account, the decryption time cost is defined as the time for the data requester to successfully decrypt the ciphertext for all levels. Figure 4b,e show the time to perform the decryption function for each scheme. Because our scheme and P-MOD follow a similar multilevel key generation approach, they are relatively convergent in terms of the time overhead of the decryption process. When measuring the decryption time cost, the time overhead of our scheme and P-MOD decreases instead as the value of k increases. This is because, with a fixed size of the shared data files, the larger the value of k, the smaller the file share of each level. From 4c,f, we can see that our scheme lags behind P-MOD in terms of the efficiency of key generation. This is due to the addition of personalized attributes to our scheme, and the values of the attributes require calls to smart contracts to complete the computation, which increases the time overhead.

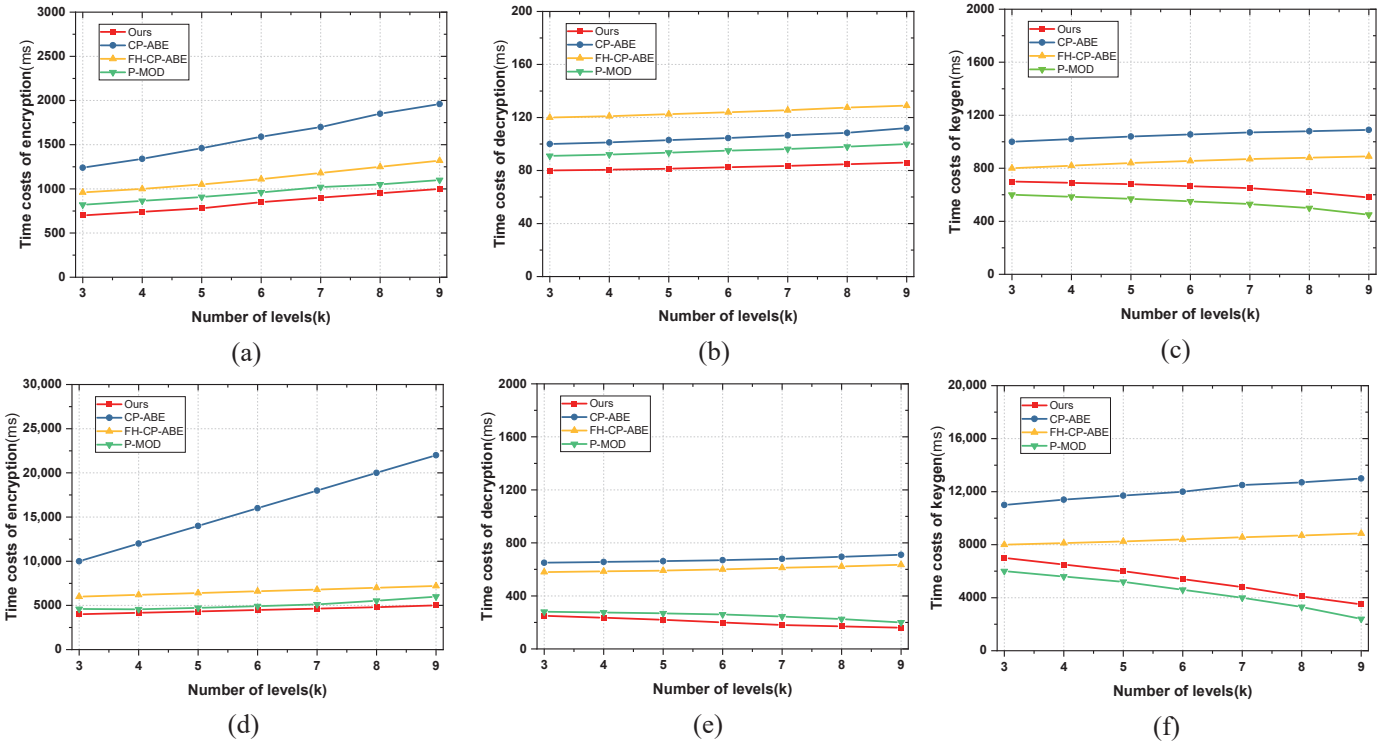


Figure 4. Time costs of encryption/decryption, public/private key generation time with the number (attrCount) of attributes and the number (k) of levels. (**a**) Time costs of encryption when attrCount = 10. (**b**) Time costs of decryption when attrCount = 10. (**c**) Time costs of public key generation when attrCount = 10. (**d**) Time costs of encryption when attrCount = 100. (**e**) Time costs of decryption when attrCount = 100. (**f**) Time costs of private key generation when attrCount = 100.

6.2. On-Chain Overhead of Our System

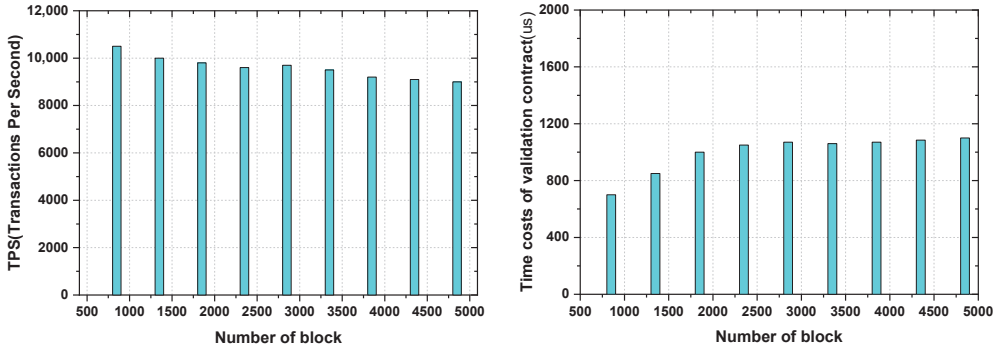
To achieve secure data sharing, both our scheme and the work in [33] utilize a blockchain to store the encrypted pointers to shared data. The difference is that we use smart contracts to verify the personalized attributes of data requesters, whereas the scheme in [33] uses smart contracts to check the satisfaction of access policies. In order to verify the feasibility of our proposed secure data sharing scheme (SDSM), we built a consortium

blockchain experimental platform (based on Hyperledger Fabric 2.2) and implemented a simulation system in this environment. The environment is configured as follows:

- The experimental network is built on a private cloud platform formed by four computing servers and one storage array.
- The experimental network is a local area network with a data transfer speed of 1000 Mbps.
- Each computing server is configured with two E5-2650v4 processors (12 cores and 24 threads each) and 128G of RAM, and the storage array is configured with 54 TB of enterprise-class SATA hard disks.
- Through virtualization technology, forty virtual servers are configured with two cores and 8 G of RAM assigned to each virtual server.
- Thirty virtual servers act as Hyperledger Fabric nodes, five virtual servers act as cloud storage servers, and the remaining five virtual servers act as attribute authority servers.

We created 100 participants with 10,000 products for a hypothetical supply chain and classified the suppliers into six classes. We randomly generated 50,000 transaction records according to a normal distribution, with 1–3 point transactions associated with each product transaction record. Both the data transactions and the point transactions are stored on the blockchain. The point transactions store the encrypted pointers to the data participants want to share, whereas the actual data to be shared is stored on the cloud platform. Each point transaction corresponds to a data sharing operation initiated by a participant and contains data ranging from 1 MB to 10 MB in total size. These data are divided into multiple files based on sensitivity, and each file can only be accessed by the partners who have reached the corresponding level. In our secure data sharing scheme, there are two functions that rely on the blockchain to be accomplished. One is that the encrypted pointer of the data shared by supply chain participants are published to the blockchain after being encrypted by attributes. The other is that the metrics of partnerships between participants require the aid of smart contracts to complete. Therefore, we completed two blockchain-related tests: (1) the throughput rate of data storage transactions uploaded to the blockchain; and (2) the average execution time of smart contracts used to validate partnerships. In our experiments, the AA acts as a blockchain node, whereas DOs and DRs act as clients.

The test results show that the processing efficiency of data publishing is stable above 9000 TPS, as shown in Figure 5a. We can also observe from the figure that the TPS decreases slightly as the submitted data increase and the number of blocks grows. This is because each of our data upload transactions contains two read and two write operations. The larger the number of blocks, the longer the read operation takes. We focus on the execution time of the validation contract because it determines the utility of our online attribute validation scheme. The validation contract mainly involves statistical queries on the size of historical transactions between participants, which is obviously affected by the growth of data volume. As shown in Figure 5b, the time consumed per query is essentially within 1 ms when the block count is small. When the number of blocks grows to more than 2000, the query time grows to more than 1 ms. Eventually, the query time stabilizes within 2 ms. From the above two tests, we can conclude that the time overhead of on-chain operations is within an acceptable time range, which indicates the feasibility of our scheme.



(a) (b)

Figure 5. Time overhead. (**a**) Time overhead for uploading data to the blockchain. (**b**) Time overhead for validating the partnership through smart contracts.

7. Conclusions

We proposed a secure data sharing scheme (SDSM) for IoT based supply chains by combining blockchain and ciphertext-based attribute encryption. The scheme supports the implementation of fine-grained access control for different levels of partnerships. There are two objectives achieved through blockchain technology. The first objective is to establish a trusted data sharing platform, where data requesters in the supply chain can verify the authenticity of the shared data. The second objective is to design a method to automatically calculate the partnership levels among participants through smart contracts based on the historical transaction facts on the blockchain. This method allows data owners to specify level thresholds to avoid arbitrariness in determining partnership levels while also reducing computational effort. We introduced personalized attributes of participants in the ciphertext-based attribute encryption algorithm and described the partnership levels by these attributes to enhance the expression of the access policy. By correlating the partnership level with the sensitivity of the data, we provide more fine-grained access control for shared data. Compared to other hierarchical data sharing schemes noted in the paper, our scheme provides an automated calculation of the data requester hierarchy. The simulations and analyses demonstrate the feasibility of our scheme. Our scheme currently only considers the case of a single supply chain, which may not be well suited for data sharing and collaboration scenarios between supply chains. In the future, we will investigate a secure data sharing scheme across multiple supply chains.

Author Contributions: Conceptualization, C.Y. and M.S.; methodology, C.Y. and M.S.; software, C.Y.; validation, C.Y.; formal analysis, C.Y.; investigation, Y.Z.; resources, Y.Z.; data curation, C.Y.; writing—original draft preparation, C.Y.; writing—review and editing, M.S. and Y.Z.; visualization, C.Y., M.S., and Y.Z.; supervision, M.S. and Y.Z.; project administration, M.S.; funding acquisition, M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by The National Key R&D Program of China (Grant NO: 2020YFB1005500) and The Leading-edge Technology Program of Jiangsu Natural Science Foundation (Grant NO: BK20202001).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, S.; Lin, B. Accessing information sharing and information quality in supply chain management. *Decis. Support Syst.* **2006**, *42*, 1641–1656. [CrossRef]
2. Lotfi, Z.; Mukhtar, M.; Sahran, S.; Zadeh, A.T. Information Sharing in Supply Chain Management. *Procedia Technol.* **2013**, *11*, 298–304. [CrossRef]

3. Tao, F.; Cheng, Y.; Xu, L.; Zhang, L.; Li, B. CCIoT-CMfg: Cloud Computing and Internet of Things-Based Cloud Manufacturing Service System. *IEEE Trans. Ind. Inform.* **2014**, *10*, 1435–1442. [CrossRef]
4. Novais, L.; Maqueira, J.M.; Ortiz-Bas, A. A systematic literature review of cloud computing use in supply chain integration. *Comput. Ind. Eng.* **2019**, *129*, 296–314. [CrossRef]
5. Arbit, A.; Oren, Y.; Wool, A. A Secure Supply-Chain RFID System that Respects Your Privacy. *IEEE Pervasive Comput.* **2014**, *13*, 52–60. [CrossRef]
6. Qi, S.; Zheng, Y.; Li, M.; Lu, L.; Liu, Y. Secure and Private RFID-Enabled Third-Party Supply Chain Systems. *IEEE Trans. Comput.* **2016**, *65*, 3413–3426. [CrossRef]
7. Hassija, V.; Chamola, V.; Gupta, V.; Jain, S.; Guizani, N. A Survey on Supply Chain Security: Application Areas, Security Threats, and Solution Architectures. *IEEE Internet Things J.* **2021**, *8*, 6222–6246. [CrossRef]
8. Cao, Y.; Jia, F.; Manogaran, G. Efficient Traceability Systems of Steel Products Using Blockchain-Based Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6004–6012. [CrossRef]
9. Wang, X.; Yu, G.; Liu, R.; Zhang, J.; Wu, Q.; Su, S.W.; He, Y.; Zhang, Z.; Yu, L.; Liu, T.; et al. Blockchain-Enabled Fish Provenance and Quality Tracking System. *IEEE Internet Things J.* **2022**, *9*, 8130–8142. [CrossRef]
10. Sun, Z.; Chen, Z.; Cao, S.; Ming, X. Potential Requirements and Opportunities of Blockchain-Based Industrial IoT in Supply Chain: A Survey. *IEEE Trans. Comput. Soc. Syst.* **2022**, *9*, 1469–1483. [CrossRef]
11. Wen, Q.; Gao, Y.; Chen, Z.; Wu, D. A Blockchain-based Data Sharing Scheme in The Supply Chain by IIoT. In Proceedings of the IEEE International Conference on Industrial Cyber Physical Systems (ICPS), Taipei, Taiwan, 6–9 May 2019; pp. 695–700.
12. Manogaran, G.; Alazab, M.; Shakeel, P.M.; Hsu, C.H. Blockchain Assisted Secure Data Sharing Model for Internet of Things Based Smart Industries. *IEEE Trans. Reliab.* **2022**, *71*, 348–358. [CrossRef]
13. Waters, B. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, 6–9 March 2011; pp. 53–70.
14. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE symposium on security and privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
15. Liu, C.; Xiang, F.; Sun, Z. Multiauthority Attribute-Based Access Control for Supply Chain Information Sharing in Blockchain. *Secur. Commun. Netw.* **2022**, *2022*, 8497628. [CrossRef]
16. Jiang, Y.; Xu, X.; Xiao, F. Attribute-based Encryption with Blockchain Protection Scheme for Electronic Health Records. *IEEE Trans. Netw. Serv. Manag.* **2022**, 1. [CrossRef]
17. Ma, H.; Zhang, R.; Yang, G.; Song, Z.; He, K.; Xiao, Y. Efficient Fine-Grained Data Sharing Mechanism for Electronic Medical Record Systems with Mobile Devices. *IEEE Trans. Dependable Secur. Comput.* **2020**, *17*, 1026–1038. [CrossRef]
18. Niederman, F.; Mathieu, R.G.; Morley, R.; Kwon, I.W. Examining RFID applications in supply chain management. *Commun. ACM* **2007**, *50*, 92–101. [CrossRef]
19. Yang, K.; Forte, D.; Tehranipoor, M.M. CDTA: A Comprehensive Solution for Counterfeit Detection, Traceability, and Authentication in the IoT Supply Chain. *ACM Transact. Des. Automat. Electron. Syst.* **2017**, *22*, 42. [CrossRef]
20. Misra, N.N.; Dixit, Y.; Al-Mallahi, A.; Bhullar, M.S.; Upadhyay, R.; Martynenko, A. IoT, Big Data, and Artificial Intelligence in Agriculture and Food Industry. *IEEE Internet Things J.* **2022**, *9*, 6305–6324. [CrossRef]
21. Piltan, M.; Sowlati, T. Multi-criteria assessment of partnership components. *Expert Syst. Appl.* **2016**, *64*, 605–617. [CrossRef]
22. Rezaei, S.; Behnamian, J. A survey on competitive supply networks focusing on partnership structures and virtual alliance: New trends. *J. Clean. Prod.* **2021**, *287*, 125031. [CrossRef]
23. Kim, J.S.; Shin, N. The Impact of Blockchain Technology Application on Supply Chain Partnership and Performance. *Sustainability* **2019**, *11*, 6181. [CrossRef]
24. Putra, F.A.; Ramli, K.; Hayati, N.; Gunawan, T.S. PURA-SCIS Protocol: A Novel Solution for Cloud-Based Information Sharing Protection for Sectoral Organizations. *Symmetry* **2021**, *13*, 2347. [CrossRef]
25. Qi, S.; Zheng, Y.; Li, M.; Liu, Y.; Qiu, J. Scalable Industry Data Access Control in RFID-Enabled Supply Chain. *IEEE-ACM Trans. Netw.* **2016**, *24*, 3551–3564. [CrossRef]
26. Qi, S.; Lu, Y.; Wei, W.; Chen, X. Efficient Data Access Control With Fine-Grained Data Protection in Cloud-Assisted IIoT. *IEEE Internet Things J.* **2021**, *8*, 2886–2899. [CrossRef]
27. Wei, X.; Yan, Y.; Guo, S.; Qiu, X.; Qi, F. Secure Data Sharing: Blockchain-Enabled Data Access Control Framework for IoT. *IEEE Internet Things J.* **2022**, *9*, 8143–8153. [CrossRef]
28. Almagrabi, A.O.; Bashir, A.K. A classification-based privacy-preserving decision-making for secure data sharing in Internet of Things assisted applications. *Digit. Commun. Netw.* **2022**, *8*, 436–445. [CrossRef]
29. Miao, Q.; Lin, H.; Hu, J.; Wang, X. An intelligent and privacy-enhanced data sharing strategy for blockchain-empowered Internet of Things. *Digit. Commun. Netw.* **2022**, *8*, 636–643. [CrossRef]
30. Jia, X.; Song, X.; Sohail, M. Effective Consensus-Based Distributed Auction Scheme for Secure Data Sharing in Internet of Things. *Symmetry* **2022**, *14*, 1664. [CrossRef]
31. Wang, S.; Zhou, J.; Liu, J.K.; Yu, J.; Chen, J.; Xie, W. An Efficient File Hierarchy Attribute-Based Encryption Scheme in Cloud Computing. *IEEE Trans. Inf. Forensic Secur.* **2016**, *11*, 1265–1277. [CrossRef]

32. Zaghloul, E.; Zhou, K.; Ren, J. P-MOD: Secure Privilege-Based Multilevel Organizational Data-Sharing in Cloud Computing. *IEEE Trans. Big Data* **2020**, *6*, 804–815. [CrossRef]
33. Zaghloul, E.; Li, T.; Mutka, M.; Ren, J. d-MABE: Distributed Multilevel Attribute-Based EMR Management and Applications. *IEEE Trans. Serv. Comput.* **2022**, *15*, 1592–1605. [CrossRef]