applied sciences

MDPI

Article

Hierarchically Authorized Transactions for Massive Internet-of-Things Data Sharing Based on Multilayer Blockchain

Shichang Xuan [1], Yibo Zhang [1], Hao Tang [1], Ilyong Chung [2], Wei Wang [1] and Wu Yang [1,*]

[1] Information Security Research Center, Harbin Engineering University, Harbin 150001, China; xuanshichang@hrbeu.edu.cn (S.X.); jackybzhang@foxmail.com (Y.Z.); tanghao@hrbeu.edu.cn (H.T.); w_wei@hrbeu.edu.cn (W.W.)

[2] Department of Computer Engineering, Chosun University, Gwangju 61452, Korea; iyc@Chosun.ac.kr

* Correspondence: yangwu@hrbeu.edu.cn

Received: 29 October 2019; Accepted: 24 November 2019; Published: 28 November 2019

Abstract: With the arrival of the Internet of Things (IoT) era and the rise of Big Data, cloud computing, and similar technologies, data resources are becoming increasingly valuable. Organizations and users can perform all kinds of processing and analysis on the basis of massive IoT data, thus adding to their value. However, this is based on data-sharing transactions, and most existing work focuses on one aspect of data transactions, such as convenience, privacy protection, and auditing. In this paper, a data-sharing-transaction application based on blockchain technology is proposed, which comprehensively considers various types of performance, provides an efficient consistency mechanism, improves transaction verification, realizes high-performance concurrency, and has tamperproof functions. Experiments were designed to analyze the functions and storage of the proposed system.

Keywords: multilayer blockchain; massive IoT data; data sharing; transaction molding; hierarchical authorization

1. Introduction

Nowadays, with the advent of the era of Internet of things (IoT), digital transformation has become an inevitable trend of social development. Big Data, cloud computing and other technologies have become the most common technology representatives, and the value of data in people's mind is becoming increasingly higher. Government departments, various types of enterprises, and individuals regard data resources as an important part of their competitiveness. The main reason is that, in the Internet of things, Big Data generated by a large number of terminal devices can make use of various data-management technologies and analysis algorithms to comprehensively process massive amounts of data, quickly acquire rules and valuable information, and realize intelligent decision-making. It has become an urgent problem for government departments and enterprises to break down data barriers and realize data sharing for better data intelligence; data sharing still faces many challenges.

With data sharing, information systems encounter a number of problems. First, data exchange is difficult because information standards and data formats are different, and ensuring the security and privacy of the transferred data is complex [1]. For example, data privacy and transparency are conflicting goals. This can result in information systems becoming information islands. The current utilization rate of massive data is low, which limits government departments and the further development of enterprises. Second, data resources owned by various entities are not equal [2,3]. Consequently, the willingness to share data is not equal. In the case of computer-threat-intelligence (CTI) sharing, most organizations are more willing to acquire CTI rather than share it, or share what

they have with only a small trusted set of entities, because the CTI is owned by these entities. Finally, most current mainstream information systems are single-data-centered platforms, managed and maintained by third-party service providers. Data service providers become authentic data owners and can perform various operations on user-uploaded data. System users have become weak parties. Therefore, most current information systems cannot provide users with a guarantee of data reliability and privacy.

The essence of this technology is decentralization that can reduce the value transfer cost and regulatory cost of traditional data-sharing platforms, and improve data reliability and transaction efficiency. Blockchain technologies offer new insight to data-sharing research. On the one hand, the application of blockchain technology in data-sharing-transaction scenarios can reduce the trust risk with decentralization, flatten the transaction process, and optimize the consensus algorithm to improve transaction efficiency. On the other hand, multiparty maintenance and distributed storage are the most distinctive blockchain features that can ensure data authenticity and reliability, and improve the credibility of data transactions.

Although bitcoin [4–11] and other blockchain application systems have been running on a global scale for a long time, and their technical value has been generally recognized, it is a pity that the function they support is still simple transactional data storage. Currently, there is no mature and stable blockchain system that can support massive data sharing on the Internet of things. Therefore, the processing capacity of blockchain technology in data-sharing applications of IoT devices needs to be improved, which is the main content of this paper.

In order to solve the problem of data authenticity in a data-sharing-transaction scenario, it is difficult to guarantee the authenticity of the system, maintenance cost is high, and privacy is difficult to guarantee. This study proposes a multilayer blockchain model that supports massive data hierarchical authorization transactions, and applies blockchain technology to data-sharing transactions in the Internet of things to provide a secure and controllable data-exchange environment. This section first introduces the architecture of the data model. Second, we outline the design of the rights-management and identity-authentication mechanism to maintain privacy and security. Finally, we used the Interplanetary File System (IPFS) protocol to separate data transactions from file storage, which improved the operating efficiency of the whole system.

2. Related Work

2.1. Internet of Things

The Internet of Things is a huge network of information sensors connected with the Internet [12–14]. Its purpose is to connect all things with the network to achieve automatic real-time object identification, location, tracking, and monitoring, and trigger the corresponding events. With the spread of mobile devices and advances in computing and ICT(Information and Communication Technology), the Internet of Everything is no longer a fantasy. In order to ensure the consistency of the development of the IoT, many researchers and institutions adopted a unified hierarchical standard. The Internet of Things can be divided into four layers: the device, network, platform, and service layers [15].

At the device layer, every-day objects have added sensors to sense the environment. The device sends the collected information to the gateway for subsequent processing and receives the response data. It is estimated that around 20 million connected devices will be online and generate data by 2020, providing unprecedented opportunities for service providers to enhance customer experience, offer game-changing innovations, and boost profits. Data no longer live in the walled garden of a traditional data center; they exist at the edges, in transit, in the cloud, and often at the middle of those paths. It may not even be clear who owns an IoT device, the rights associated with the data, and who is responsible for ensuring the integrity and security of the data in question. When these fundamental

issues are not addressed, it is difficult for service providers to provide clarity and transparency to customers about their data practices.

Faced with these problems and challenges caused by the huge amount of data generated by IoT devices, many industry insiders have started to turn their eyes to this place for research. Companies such as Microsoft, Bosch, Samsung, and Fujitsu are already collaborating on a blockchainlike market for Internet of Things data, which they hope will one day allow any connected sensor or device to independently and cheaply obtain data. If successful, the market could provide powerful applications for supply chains, smart cities, artificial intelligence, and other use cases. If done well, such markets could illustrate how companies can better balance the twin requirements of IoT data security and open data sharing.

In general, in the context of the Internet of Things, there are a large number of Internet of Things terminal devices generating a large amount of data. In addition, Internet of Things devices themselves have limited capabilities and poor attack resistance, resulting in extremely low security of a large amount of data generated in the process of transmission, exchange, and sharing. How to effectively solve this problem is the focus of this paper.

2.2. Data Sharing

With the booming development of the Internet of Things industry, data are becoming a valuable asset in our society and economy. In various fields of the Internet of Things, such as smart transportation, smart medical care, and smart homes, a large amount of data and data exchange are generated. As a result, the data-exchange market is becoming increasingly popular. In data-exchange markets, data owners can share or sell their data to data consumers. In smart medicine, for example, sharing patient data between different hospitals can help nurses and doctors judge patients' health and make real-time decisions, even in remote areas. However, the current data-exchange market is centralized, and all participants must trust authorized third parties. In centralized markets, data owners and customers pay a fee to authorized third parties. In addition, a single point of failure may occur. To overcome these challenges, data owners and customers can collaborate to build decentralized data-exchange markets using blockchain technology. Aiming at personal data sharing under personalized service, a study [16] proposed an anonymous dataset-distribution scheme based on blockchain. The platform consists of peers acting as data and receivers, and transaction validators. The blockchain is used to record all transactions of anonymous datasets between data agent and receiver, and to meet the security requirements of data distribution.

Aiming at the difficult issues of copyright and privacy protection in the current data-exchange market, another study [17] proposed a distributed Big Data exchange solution. The solution aims to foster an ecosystem in which all participants can collaborate and exchange data in a peer-to-peer manner. The core part of the solution is to use blockchain technology to log transactions and other important documents. Unlike existing data-exchange markets, their solution does not require any third party. It also provides data owners with a convenient way to audit the use of data to protect data copyright and privacy.

In patient health systems with personal data sharing in different medical systems, running the risk of privacy breaches, a study [18] proposed an application (medical-data gateway (HDG)) on the basis of the blockchain for patients to control and share their own data easily and safely without violating their privacy. This provides a new potential method to improve the intelligence of healthcare systems while keeping the patient data private, ensuring that patients own and control their medical data, and trusted third parties can utilize patient data when privacy is not invaded.

The main objective of [19] was to describe a method to effectively and securely share medical information via a data-sharing network. Patient records should be consistent and cross-institutional boundaries, with access conditions strictly at the discretion of the patient. As a second goal, not only should these data be shared, but also shared in a way that all interested parties can understand their structure and meaning, ultimately improving data utilities and patient care. In the process of

medical-care-data sharing, securely sharing health data with other nodes in the network by PSN nodes is a medical-security system based on VPN. The authors in [20] designed two protocols for this system. The first is an improved version of the authentication association shown in IEEE 802.15.6. It establishes secure links with unbalanced computing requirements for mobile devices and sensor nodes with limited resources. The second protocol uses blockchain technology to share health data between PSN nodes. We implemented a protocol suite to study protocol runtime and other factors. In addition, in some use cases, a human body channel is proposed for PSN nodes. The proposed system illustrates a potential approach to implement PSN-based applications using the blockchain; data are the core issue. Table 1 summarizes the above blockchain-based solutions. In addition, we compare the advantages and disadvantages of the above blockchain-based solutions in Table 2.

Table 1. Data sharing of blockchain-based solutions.

Ref.	Blockchain Data	Contributions
1	Transactions between data brokers and receivers	Anonymized dataset-exchange platform without centralized trusted third party
2	Transaction logs between data owners and consumers	Blockchain-based data-exchange system that enables all participants to exchange data in a peer-to-peer way
3	Personal medical data	Medical-data gateway (HDG)-centric healthcare architecture that enables patients to securely manage and control their own medical data
4	Healthcare data	Blockchain-based approach to effectively and securely share healthcare information among institutions
5	Healthcare data	Blockchain-based healthcare system to enable secure health data sharing among PSN nodes

Table 2. Advantages and shortcomings of blockchain-based solutions.

Ref.	Advantages	Shortcomings
1	Prototype of proposed platform implemented on the basis of Hyperledger Fabric.	Economic model for proposed anonymized dataset-exchange platform not considered.
2	Blockchain applied to record transaction logs between data owners and consumers.	Prototype of proposed system not implemented.
3	Blockchain used to store personal medical data securely and immutably.	Consensus algorithm and incentive mechanism not considered.
4	FHIR chosen as sharing format of electronic health records.	Robust Master Patient Index (MPI) approach to consistently identify a patient among institutions not provided.
5	Improved protocol based on IEEE 802.15.6 proposed to establish secure links between sensor nodes and mobile devices.	Performance of large-scale PSN-based healthcare system not tested.

It can be seen that, in the context of the Internet of things, existing data-sharing schemes based on blockchain are only at the idea level, and there is no system that can comprehensively consider throughput, delay, privacy protection, and other factors.

2.3. Blockchain Background

Blockchain technology was proposed in Satoshi Nakamoto's important paper "Bitcoin: A Peer-to-Peer Electronic-Cash System" [21] (blockchain 1.0). A blockchain can be simply understood as an unchangeable transaction book. Transaction data are stored in a contiguous data block in the order of timestamps, ensuring data integrity and privacy through the use of cryptography and distributed techniques. Due to the wide application scenarios of blockchain technology, many large organizations have satisfied their different needs by improving their business models or operating

modes, and have achieved many new features. So far, the development of blockchain technology has gone through four milestones, as shown in Figure 1.

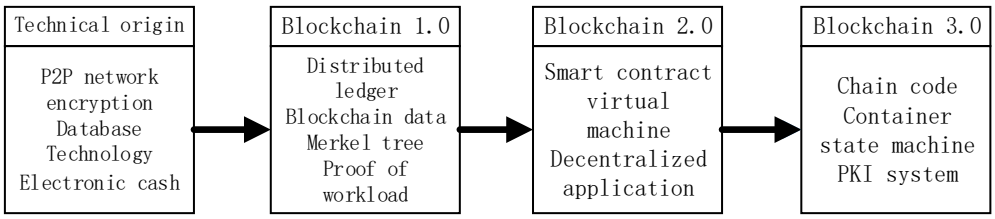

Figure 1. Blockchain-technology development stage.

At present, current research on blockchain technology focuses on its data model, block-structure design, consensus algorithm, and smart contracts. The development stage of blockchain-technology milestones is as follows. In December 2013, Vitalik Buterin proposed ethereum blockchain technology [22–27] (blockchain technology 2.0). Ethereum, as another form of digital currency, was born. Compared with Nakamoto's bitcoin, ethereum technology provides users with a complete Turing scripting language and proposes the concept of smart contracts [28], creating a decentralized, automatically maintained, and reliable distributed network. In December 2015, led by the Linux Foundation, IBM, Intel, and Cisco jointly announced the establishment of the Hyperledger Joint Project (blockchain 3.0). For the first time, the superbook used blockchain technology in the application scenario of distributed ledgers, providing open-source project design with open and transparent decentralized features, and opened a new beginning for building an efficient business-transaction network [29].

With the continuous development of blockchain technology, the application scope of blockchain is very wide and can be roughly divided into three types: public blockchain, private blockchain, and consortium blockchain.

Public blockchains: blockchains that any organization structure or individual can join, read data from, form a consensus, participate in transactions, and get effective confirmation. As an alternative to the current mainstream centralized-trading platform, the security of the public chain is maintained by the "consensus mechanism". The consensus mechanism can adopt PoW(Proof of Work) or PoS(Proof of Sstake)to encourage mining nodes to produce blocks in the form of economic rewards. In the whole consensus process, the more blocks are dug out by nodes, the more economic rewards they get.

Private blockchains: A private-blockchain system is generally deployed in the internal body of a specific organization. Different from public chains, the read and write permissions of the blockchain system are set by the affiliated organization, and not all people can participate in the private chain network. Internally, each node is decentralized. Compared with a public chain, the great advantage of private chains is that, in P2P network systems, the node with the lowest processing performance often determines the processing speed of the whole system. However, since the private chain is only controlled by an organization, all nodes and the network environment of the system are completely controllable, so it can be ensured that the private chain is far superior to the public chain in processing speed.

Consortium blockchains: There are many preselected nodes in the consortium blockchain, and they play a dominant role in the process of consensus formation. An alliance chain is usually applied in the organization alliance of a specific field, and each node is usually a specific organization subject that can only join or quit the alliance chain after being authorized by the authoritative organization. In terms of access to transaction information, the alliance chain sets multiple permissions, which may be public or only allow the two parties involved in the transaction to read. The alliance chain is characterized by multicentralization, strong controllability, nondefault data disclosure, and fast transaction speed.

In terms of blockchain block structure and chain model, Yuan Yong systematically introduced the hierarchical structure and data model of blockchains, and explained in detail their basic theory, technical methods, and application scenarios, providing useful information and references [30]. The proposed bitcoin-NG(A block chain protocol for serializing transactions) model added key blocks and microblocks that improved block storage capacity and data-transmission throughput. Overall, the blockchain model improved the performance of data transactions and optimized the blockchain. Factom [31] layered the blocks. A microchain consists of hierarchical blocks, and a hash-calculated reference is stored on the chain. The layered architecture maintains the data throughput of blockchain transmission data under the premise of consistency [32]. Cai Weide proposed the design concept of the Beihang chain, expounding the double-chain model: ABC and TBC, in which the ABC chain is the account blockchain that is responsible for the inquiry and preservation of the book data; and the TBC chain, which is the transaction blockchain that is executed between the main institutions. This double-chain structure model improves system throughput, reduces delay, and enhances privacy protection, which provided a theoretical basis for our research.

Existing blockchain platforms and related information are shown in Table 3.

Table 3. Blockchain platforms and related information.

Platform Name	Type	Data Model	Consensus Algorithm	Digital Currency
Bitcoin	Public chain	Transaction data	PoW	Bitcoin
Ethereum	Public chain	Account data	PoW/PoS	Ethereum
Hyperledger Fabric	Alliance chain	Account data	Solo/Kafka/SBFT (Scalable decentralized trust infrastructure for blockchains)	-
Sawtooth	Public chain/union chain	Account data	PoET(Proof of Elapsed Time)	-
BigchainDB	Alliance chain	Transaction data	Quorum Voting	-

2.4. IPFS Protocol Background Knowledge

This part first analyzes the shortcomings of the current mainstream hypertext transfer protocol (HTTP), and introduced the origin of the development of the IPFS protocol. Second, the protocol concept and technical framework of the IPFS protocol are elaborated in detail. Finally, the operational process of the IPFS protocol is analyzed and studied. This provides a theoretical basis for the application research of the IPFS protocol in Section 3.

(1) HTTP Disadvantages

Highly centralized: The original intention of web designers was to decentralize and realize an interconnected world, but the current situation is that 80% of people rely on 20% of central services, and the normal operation of an HTTP protocol largely needs to be supported by a central network with excellent performance. Many organizations can easily monitor request data by only deploying a few nodes in a communications network, and can easily block websites from accessing centralized resources. HTTP communication networks are also more vulnerable to DDoS(Distributed Denial of Service attack) attacks and thus face great risks.

Low efficiency: It is too expensive to always distribute data from central facilities if they are not already processed. If you do not need a central database, you can greatly reduce the transmission consumption of network resources by transferring the required data from nearby nodes.

(2) IPFS Protocol Design Concept

The IPFS data-transmission protocol relies on a P2P network to realize distributed storage. Its original intention is to supplement or even replace the current mainstream HTTP protocol, and its

purpose is to build a faster, safer, and free Internet. The prominent feature of the IPFS protocol is a radical change in the way data files are searched. Compared to the HTTP protocol, this protocol is no longer concerned about the location of the central data-storage server, and the file name and storage path on the server, but only the content of the file. The resource file is stored in the IPFS protocol network, and the hash value, generated after the hash-function processing, can represent the uniqueness of the file data. The IPFS protocol has virtually no storage restrictions. Larger files are subdivided into smaller chunks. When a client downloads a file, it can obtain smaller chunks of file data from multiple servers at the same time. This design can provide storage services for all kinds of data, including document data, images, videos, and operating systems.

In conclusion, hierarchically authorized transactions for massive data sharing based on multilayer blockchain are proposed based on the shortcomings of some existing data-sharing schemes. Data can be shared efficiently and with privacy. At the same time, when designing data storage, the system adopts the IPFS [33] protocol to separate data transactions from file storage, improving the operating efficiency of the whole system.

3. Multilayer Blockchain Model Supporting Massive Data Hierarchical Authorization Transactions

3.1. Overall Design

The overall architecture design of the blockchain data model is shown in Figure 2: root certification authority (RCA) of the alliance chain, the network deployment diagram of the nodes in the alliance chain and the internal nodes of the main body, the data-storage diagram, and the blocks of the alliance- and private-chain structure.

As shown in Figure 2, there is an RCA in the alliance chain. Each private-chain node has its own certificate authorities (see below). These certificate authorities use certificate trust chains to ensure the legality of authorization at all levels. The alliance-chain block is responsible for transactions and stores transaction information. The certificate trust chain is an ordered list of certificates containing SSL and certificate authority (CA) certificates, enabling the receiver to verify that the sender and all CAs are trustworthy. The chain or path begins with an SSL certificate, and each certificate in the chain is signed by the entity identified by the next certificate in the chain. The chain terminates at the root CA certificate. The signatures of all certificates in the chain must be verified up to the root CA certificate, which is always signed by the CA itself. The private-chain block is responsible for data storage and for storing (the hash of) data files. The actual storage of the data files is under the chain, and files are actually stored in the IPFS network built by the alliance's organization. The world state of the books is stored in a LevelDB database (world state is a state tree maintained by all nodes that join the blockchain).

3.2. Data-Model-Architecture Design

3.2.1. Multicenter Blockchain Layered Structure Design

Through the research and improvement of the technology principle of alliance-chain Hyperledger Fabric, this section proposes the hierarchical-structure design of the multilayer blockchain as shown in Figure 3.

(a) Interface Layer

This layer provides the external service interface of the data model and provides functions such as querying transaction information and data ID, initiating transactions, registering identity information, and obtaining transaction certificates; it also supports the use of smart contracts for secondary development. The interface to the smart contract is provided by the Fabric system to help users develop smart contracts.

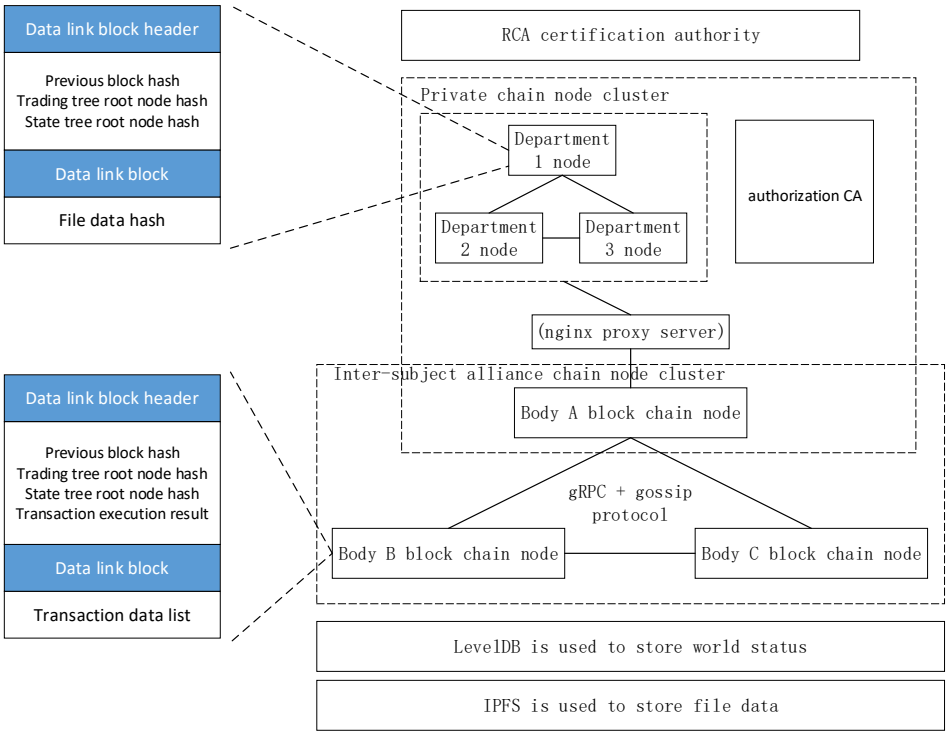

Figure 2. Overall data-model-architecture design.

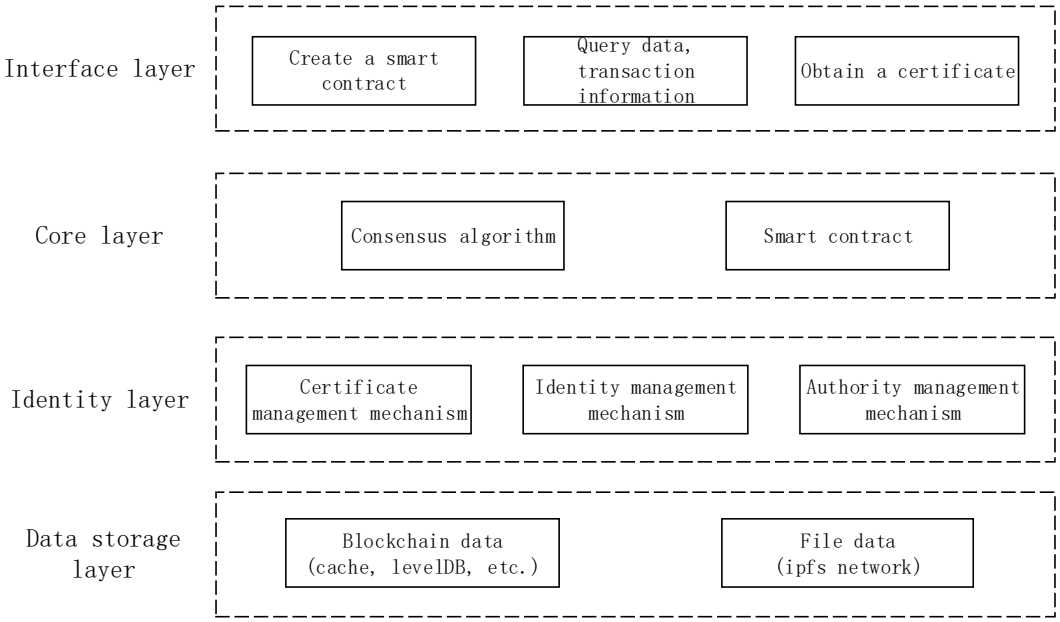

Figure 3. Data-model hierarchy.

(b) Core Layer

The core layer is divided into consensus algorithms and smart-contract services. This layer accelerates the formation of data consistency in distributed systems through the setting of algorithms and strategies, and by accelerating transaction speed and improving throughput. At the same time, smart contracts provide deployment, invocation, and query services for the writing of interface-level smart contracts.

(c) Identity-Authentication Layer

This layer is responsible for identity authentication and permission control. The certificate-management mechanism issues a certificate for newly joined nodes by establishing a CA chain to ensure that nodes joining the alliance-chain network are secure and reliable . The identity- and rights-management mechanisms provide privacy protection for IoT transaction data to ensure operational security and reliability.

(d) Data-Storage Layer

This layer is responsible for data storage on the blockchain. These data are important information about the header and body of the block, and are generally stored in memory. The amount of data is small. The world state of the blockchain is stored in a database similar to levelDB, and the data stored in the block remain in key-value form. A large number of Internet of Things data files need to be stored in the IPFS network. The stored file data can be queried in the IPFS network by the file ID stored in the block. This storage method ensures data consistency between nodes. This section focuses on the interface, authentication, and data-storage layers. The fourth section describes the core layer in detail.

3.2.2. Chain Construction Design

This model is based on the alliance-chain design, and the alliance chain is applicable to data-sharing transactions between multiple organizations. Building a private chain inside the main body is used for storing data information. The double-chain construction model is shown in Figure 4.

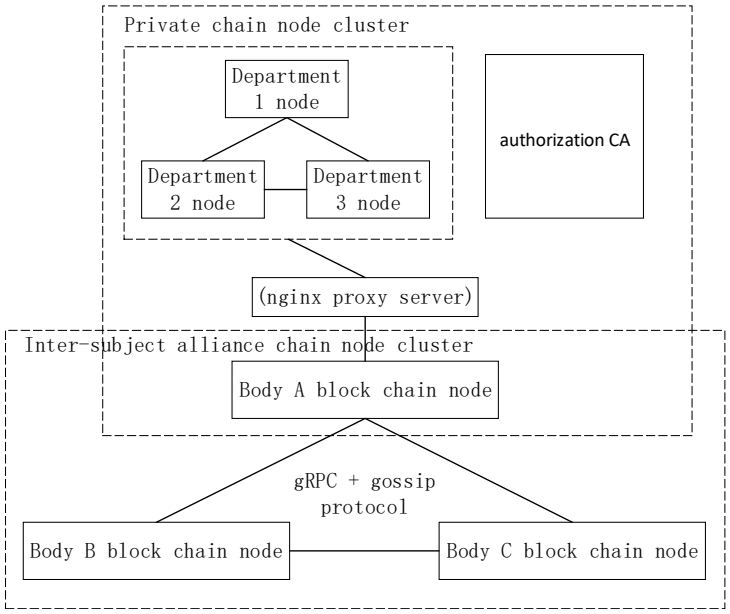

Figure 4. Overall data-model-architecture design.

As shown in Figure 4, the double-chain model is composed of a chain of alliances and a private chain. In the alliance chain, each organization or body is taken as the center to macroscopically form a multicenter blockchain structure. First, a private-chain network inside every organization is built. The organization can have different departments as nodes of the body cluster. Internal data-information sharing can be achieved through the private-chain body. Due to the nature of chain blocks, this can ensure that data are not tampered with and are fault-tolerant. Centralized data storage and fault-tolerant server-maintenance costs, on the other hand, are too high. At the same time, the private chain is deployed in a local area network and isolated from the Internet, so the security of the private data inside the main body can be guaranteed by physical isolation. An identity CA is used to provide identity certificates for nodes participating in the transactions, while a transaction CA is used to verify the identity of the counterparty and the legality of the transaction. A proxy server separates the alliance chain and the private-chain network. The proxy server to the blockchain node of the main

body transmits the data-transaction information, and transactions are performed between alliance chains.

The alliance chain consists of a principal node and a CA server node. The identity CA is responsible for reviewing the identity of the organization and performing operations such as certificate issuing and revocation. The principal node is responsible for trading between the main body, the alliance chain is responsible for storing the transaction data, and the transaction initiator has the right to conduct related transactions after being authorized by the transaction CA. When the transaction is initiated, the transaction-verification and consensus-sorting steps are also required. After the transaction is completed, the node transaction information is stored. For billing processing, this blockchain model supports the gossip network protocol, and each node communicates via gRPC.

3.3. Business-Intelligence-Contract Design

This section introduces the business intelligence contract design of the private chain and alliance chain of this blockchain model. In a conventional data-sharing blockchain model there is a single chain, and data files, smart contracts, and transaction information are placed together on this chain. All data can be seen by all organizations participating in the data transactions on the chain. This is not in line with the business needs of the actual data-sharing applications, because most organizations do not want data of their own transactions to be seen by all other organizations. Consequently, privacy protection was added to our design. We propose a multilayer blockchain architecture model. The previous section analyzed the hierarchical structure and architecture design of the data model. This section focuses on the business processing of the two types of chains, and separately analyzes the dual-chain services.

In the private chain, data-file information is stored without participating in the transaction, so the contract design of the chain is for the operation of the data file, including operations such as adding, deleting, modifying, and querying.

3.3.1. Private-Chain Business-Intelligence-Contract Design

The private chain only stores data files and user information, does not execute transactions, is built inside the main body, uses the internal local area network (LAN) of the main body to communicate, and is isolated from the external Internet to ensure the security of the main data. The platform uses the IPFS protocol to store real files. Private-chain blocks store the hash values of the files processed by the IPFS technology. The IPFS client or browser can obtain file data according to the hash value. Subsequent sections describe this content in detail. This is currently understood as a private chain capable of carrying data files. The business intelligence contract of the private chain is designed as follows: First, a user can upload data files through a private-chain system. Since data-file information is stored on all nodes of the network through the platform, the model can ensure that it is difficult to tamper with data information. Users can query, add, delete, and modify file-data information through smart contracts. At the same time, distributed data-storage methods can enhance the system's disaster tolerance and reduce system-maintenance costs. The data designed by the business-intelligence contract are divided into user information, data information, and operational information. Details are in Table 4.

Table 4. Data-structure design of private-chain smart contract.

User Information	Data Information	Operational Information
Department name	Data ID	Numbering
Account public key	File account	Account address
Account private key	-	Generation time
Account address	-	Operation type:

Data ID is file hash based on Interplanetary File System (IPFS). Operation type: 0 = query, 1 = add, 2 = delete, 3 = modify.

3.3.2. Private-Chain Business-Intelligence-Contract Design

It constructs the alliance-chain structure from each organization, and it is responsible for data-sharing transactions between organizations. The alliance chain is composed of the nodes of each body and is the channel for data transaction and acquisition between bodies. On the basis of characteristics of the alliance chain, the blockchain system applies the channel mechanism to implement access control. The body in the same channel sends the data information of the transaction in encrypted form, and the transaction data are only stored in the account of the node of the same channel. Only groups involved in the transaction can obtain transaction-data information.The data structure design of the alliance chain is shown in Table 5.

Table 5. Data structure design of alliance-chain smart contract.

Trading Information	Operational Information
Initiating party address	Id
Accept transaction-party address	Account address
Transaction data ID	Generation time
transaction hour	Operation type
Trading status	Trading status

Transaction data ID based on IPFS-generated file hashes. Operation type: 0 = query, 1 = add, 2 = delete, 3 = modify. Trading Status: 0 = trading new release, 1 = transaction verification successful, 2 = transaction verification failed.

3.4. Authority-Management and Identity-Authentication Mechanism Design

This section describes the design of the rights-management and identity-authentication mechanisms provided by this blockchain model. We introduced a PKI system to achieve identity management, set up alliance-chain–private-chain CA servers at all levels, issue or revoke digital certificates for entities and nodes participating in or exiting from the network and transactions, and verify their identity information and transactions. We introduced the channel mechanism to ensure the privacy of transactions by putting transacting nodes into the same channel for each transaction. In the transaction process, only the department of the same channel can see data associated with the transaction, and transaction information is only stored in the nodes of the participating transaction. This mechanism provides strong privacy and security for each transaction.

3.4.1. Overall Design of Certification Mechanism

The blockchain model realizes the supervision of user transactions by issuing identity-authentication certificates (ICerts) and transaction-authentication certificates (TCerts) to users. This enhances the credibility of the user node by ensuring the privacy and security of the data transactions through background supervision of the double-blind transaction function. Before a user can access the blockchain system, the identity-certification authority (ICA) of the group to which the user belongs is responsible for registering the identity of the user and issuing an ICert to them. This information is synchronized with the RCA certification of the alliance-chain network. When the user needs to make a transaction, the transaction-certification authority (TCA) is responsible for issuing the TCert and synchronizing this information with the RCA certification authority of the alliance-chain network. The TCert is first signed by the (private key of the) certificate of the user who publishes the transaction, which hides the real identity of the user and realizes the pseudonymity of the transaction. The CA infrastructure architecture of the blockchain model is shown in Figure 5:

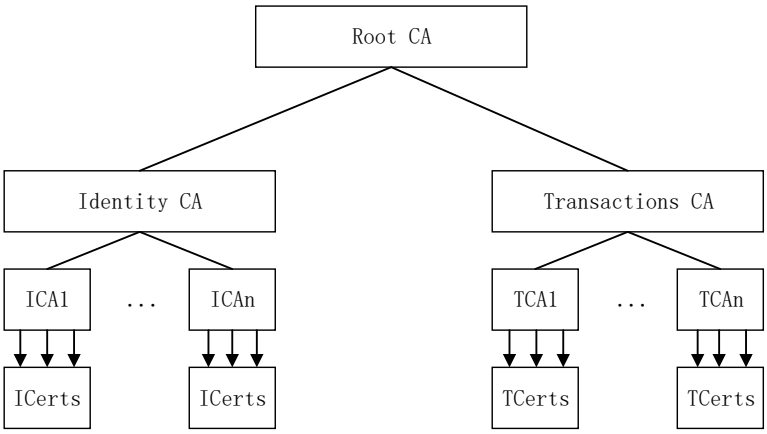

Figure 5. Certification-authority (CA)-infrastructure architecture of blockchain model.

In this blockchain system, there is only one root certificate that is deployed in the federated chain network. The RCA is responsible for certifying the CAs of the participating organizations. Each organization internally deploys a CA server, which is divided into ICA and TCA functionality, and is responsible for issuing certificates (ICerts and TCerts) for node authorization within the body. CA can authorize the node of this body to make transactions for other bodies, and the authorized CA is mainly responsible for the management of identity and transaction certificates within the main body, including registration, clearance, and other operations. In the transaction process, there are nodes to check whether the relevant transaction conforms to the certificate authority, including authentication, transaction inspection, and other operations. Data transaction continues after the check is correct. Through identity and transaction authentication, privacy protection in the multicenter blockchain model can be guaranteed, and the node can communicate with other bodies.

3.4.2. ICert Mechanism Design

The ICA [34] issues ICerts to all nodes within each group, which are long-term certificates and can be revoked once registered. ICerts provide authentication for the registration of TCerts during the transaction-certificate-request process. An ICert includes the registration number of the certificate owner and two public keys (the signature key and the encryption key). By obtaining the ICert, the TCA can perform corresponding transaction verification. The ICert process is shown in Figure 6. The ICA is responsible for issuing the ICert to each new user. They are registered into the system for identity verification.

3.4.3. TCert Mechanism Design

TCA issues TCerts by querying the user's ICert. As shown in Figure 7, the user first inquires through the client whether they have a TCert certificate. If the TCert does not exist, then the application for it is conducted through the TCA. Before this, the current user needs to have an ICert in order to first perform authentication. After the ICert is verified, the TCert is issued. Once authorization is complete, the user can either work on the transaction or directly store the TCert in the database. Users need to have different TCerts to participate in different transactions.

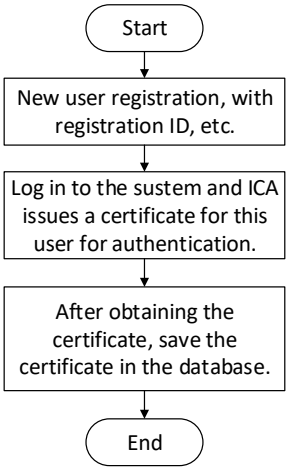

Figure 6. Identity-authentication certificates (ICert) mechanism flowchart.

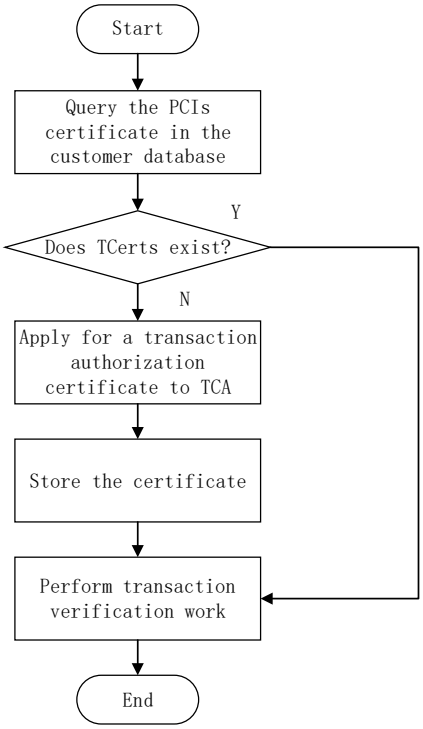

Figure 7. Transaction-authentication certificate (TCert) mechanism flowchart.

3.5. Privacy Access Control Design

Taking a transaction between Organizations A and B as an example, Organization A initiates a data transaction to organization B. Organization B obtains the data content through the transaction. However, the organizations involved in the transaction want this to be done in private and do not wish other users of the platform to view them or the transaction information, or obtain the data content. Therefore, the transaction cannot go through the process of reaching consensus across the entire network. Therefore, Organizations A and B can only reach a consensus between themselves through the channel mechanism. This is done by establishing a consensus service node independent of the organizations. Adding Organizations A and B, and the consensus service nodes to the channel, consensus ordering is conducted by the consensus service node for the transactional data between Organizations A and B. The data information for the transaction is stored only in the books of Organizations A and B. Through this method, external organizations cannot see the specific transaction

information between Organizations A and B, and the access control function is realized. The schematic diagram of the channel permission control mechanism is shown in Figure 8.

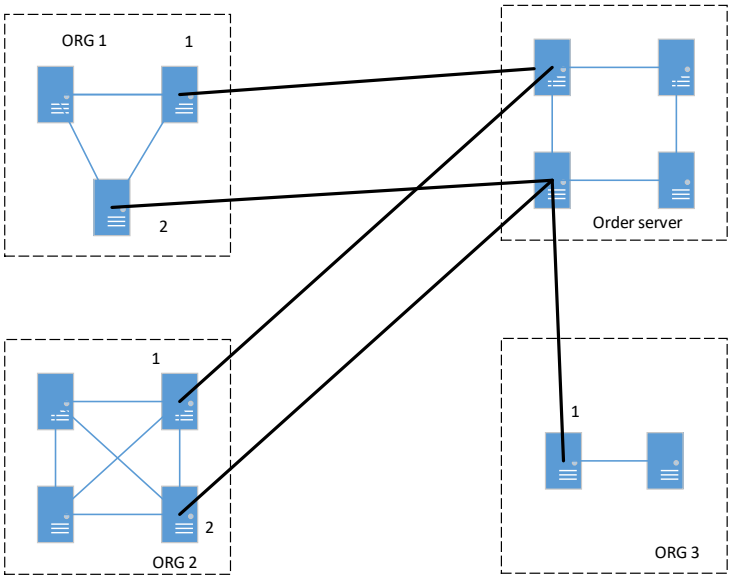

Figure 8. Schematic diagram of channel permission control mechanism.

The thick black lines shown in Figure 8 are the connections of the channel pipelines. There are two transactions: Transaction 1 contains node Peer1 at ORG1 and peer1 at ORG2. Transaction 2 contains node Peer2 at ORG1, Peer2 at ORG2, and Peer1 at ORG3. These two transactions are isolated from each other by the system. In the blockchain network, the same node of the same body can participate in different channel pipelines, and the same channel can be joined by different nodes of different organizations. These transactions can only be seen by nodes joining the current channel pipeline, and other nodes cannot see the relevant information of the transaction. Nodes of the same channel generate the same accounting information, and nodes of different channels have different accounting information. The channel mechanism relies on the consensus service algorithm to realize the privacy access control of the platform. By dividing the multicenter blockchain model into multiple logical channels, nodes participating in different transactions are included in different channels, which is well-adapted to data-sharing transactions.

The implementation of the channel mechanism also depends on support provided by the message queue. The topic queue and the broadcast mechanism of the message queue can handle the channel function well, and nodes of the same channel send the same accounting information to the consensus service node through the message queue. The corresponding relationship between multichannel and ledger is shown in Figure 9.

As shown in Figure 9, there is a one-to-one correspondence between channel and ledger. In the figure, the Peer1 node joins Channel1, Channel2, and Channel3, so the books of the three channels are maintained in the Peer1 node. The Peer2 node has joined Channel1 and Channel2, so the Peer2 node maintains the ledger of the two channels, and so on. The channels and accounts that the node joins are one-to-one. This design enables data transactions to be performed simultaneously for each node of each body during the transaction process, and data transactions of all nodes in all channels can also be concurrently performed. Compared with the usual single-chain structure platform, this design can realize transaction parallelization under the premise of ensuring transaction data privacy, improved data throughput, of the whole network, and faster transactions.

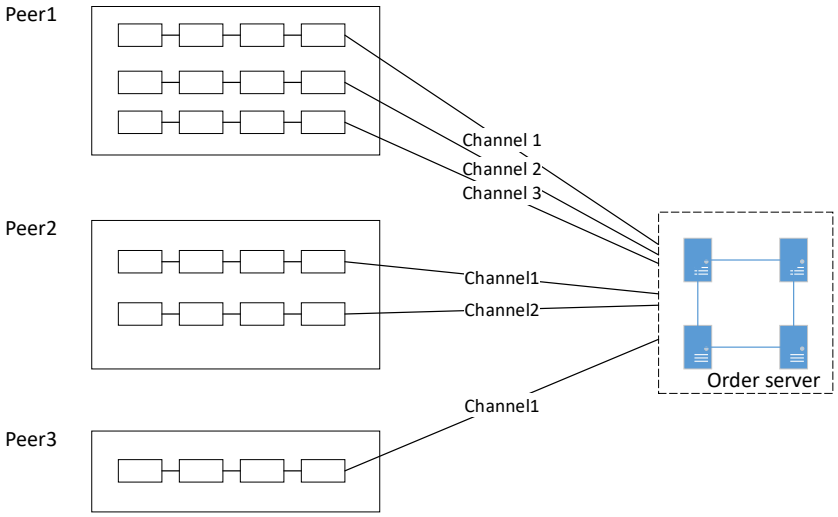

Figure 9. Schematic diagram of correspondence between multiple channels and peers.

3.6. Data-Storage Structure Design

Blockchain technology is applied to data exchange in the Internet of Things to generate and process a large amount of data. However, in a traditional blockchain system, each node must be able to handle all transactions and maintain the entire transaction back to the first block (the genesis block). Therefore, blockchain technology cannot be directly applied to IoT data exchanges with limited device-storage resources. The general idea to solve the storage problem is to combine the blockchain with existing P2P storage or database so that it can store a larger amount of data. Therefore, this paper adopted a storage mode combined with the IPFS system.

The data structure related to the data book is constructed. A ledger is a series of backup logs based on transaction actions that are sorted and are difficult to be modified. The trading action refers to the execution of the chain code to generate the transaction result. Each transaction provides a series of transactional accounting information through corresponding operations such as adding, deleting, and querying, and the information is recorded on the ledger in the form of a key-value pair. By using the consensus algorithm to package a certain number of transactions into blocks, this enables the books to be connected in a blockchain. At the same time, there needs to be a database, called the world state, to maintain the state of the ledger. The following describes the storage structure of the book and the storage contents of the world-status database.

The world-status database depends on the file system. Each specific block is stored in a file. The LevelDB database stores the file ID corresponding to the transaction and the offset of the file data. The function of LevelDB is to add an index to the book to facilitate the quick querying of the transactional information.

The world-status database stores the latest key-value pairs generated by smart contracts during the transaction process, which can be understood as the latest transaction information. Calling a smart contract for trading can change state data and store the latest key-value pairs in the world state database, which can improve the efficiency of smart contract calls.

The block structure of the ledger is shown in Figure 10. The block head of the blockchain system generally includes the previous block hash, transaction tree hash, state tree hash, and transaction-execution results, which are recorded in the form of a key value. The block body mainly stores transaction records, including transaction and contract ID, public key and signature of the parties, and intelligent contract.

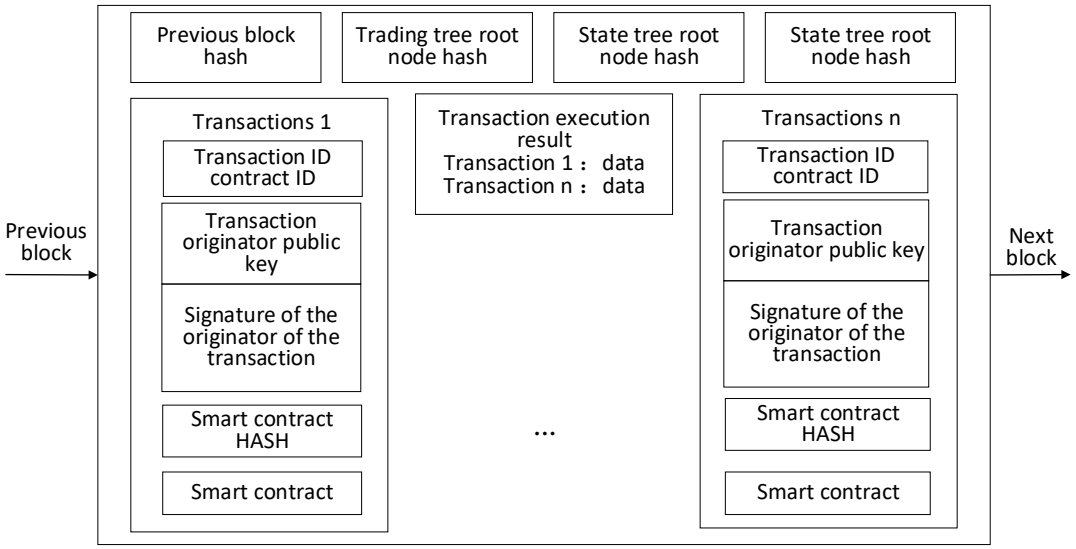

Figure 10. Schematic diagram of correspondence between multiple channels and peers.

The process of obtaining data files is shown in Figure 11. This blockchain system adds the hash of shared data files into the transaction content, and real files are stored in the IPFS private network built by the alliance. The user logs into the IPFS client or browses through the file hash, and the CA in the alliance-chain network audits the ICert of the relevant user to determine whether to authorize the user to access the file content after the user participates in the transaction.

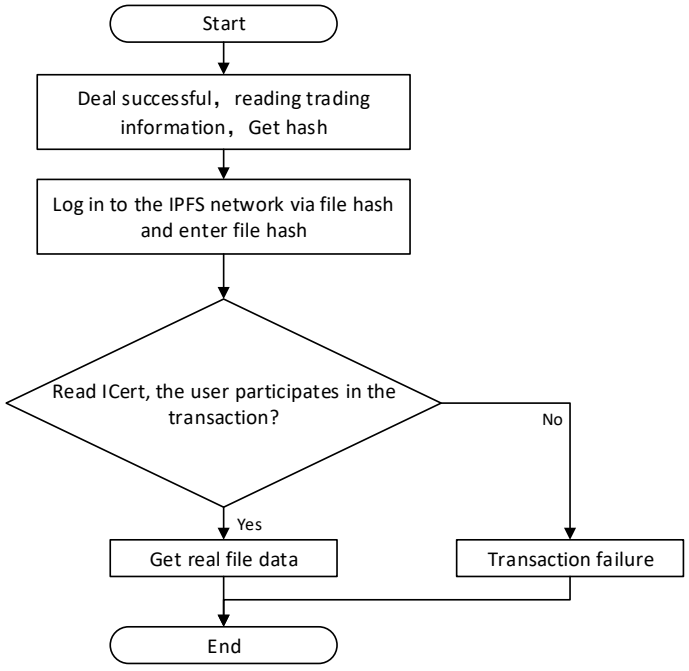

Figure 11. IPFS network data acquisition flowchart.

3.7. Consensus-Algorithm Design

In order to adapt to the blockchain model proposed in this paper, we also designed an efficient consensus algorithm to match it and achieve the purpose of trade parallelization and fast processing.

By designing this consensus algorithm, the parallel processing of transactions can be realized in the data transaction of Internet of Things devices, which can improve throughput and speed up the transaction.

According to the actual characteristics of our proposed multilayer blockchain model, the required consensus algorithm should have high throughput, low latency, high transaction efficiency, and low power consumption as its design principles.

3.7.1. High Throughput and Low latency

High throughput and low latency are important indicators to evaluate the merits of a consensus algorithm. High throughput means that the platform system can process more transaction data over a certain period of time. Low latency means that less time is spent between initiating a trade and confirming an exchange. These two points are mainly related to transactions, and the key problem to be considered in designing a consensus algorithm is how to improve the speed of transaction-data transmission and transaction verification. Therefore, it is most important to design a consensus algorithm with high throughput and low latency.

3.7.2. Low power consumption

Since the platform system was designed on the basis of blockchain technology, and blockchain is based on the distributed concept, participating in consensus in the distributed system requires the joint participation of multiple nodes, so it is more energy-consuming than a traditional central-platform system. Therefore, it is very important to design a consensus algorithm with relatively low energy consumption. Take bitcoin consensus algorithm PoW as an example. The general idea of this algorithm is that a miner node continuously calculates random values through hash functions and finally obtains the hash value that meets the requirements. The miner node that calculates the result first gains the right to block. From this, we can see that the PoW consensus algorithm requires the miner node to perform repeated hash calculations all the time. It was proven that, although this consensus algorithm guarantees extremely strong security, its annual power consumption is equivalent to the annual power consumption of a country [35], so this algorithm is very resource-consuming.

Through the above analysis, we can set weights to evaluate the merits of the consensus algorithm; the weight formula is as follows.

$$value = x \times tps + y \times energy \tag{1}$$

Since the alliance chain is responsible for trading, according to the design principle of consensus algorithm, this algorithm was optimized in dealing with trading speed and reducing energy consumption. The introduction of the message-queue and book-storage technology because the same node can join different channels in a transaction at the same time, a trade-agreement-generated book number depends on the number of the channels. As a result, the algorithm can implement trade parallelization to improve system throughput and speed up the transaction confirmation time. At the same time, this algorithm abandons the traditional way of acquiring accounting rights through comparative computing power, increases the endorsement node's participation in consensus, accepts the transaction data of message queue, sorts the transaction data, and forms the ledger data. Compared with PoW, this method can greatly save power.

The nodes in the alliance-chain network are collectively referred to as peer nodes, and a peer may play different roles in the consensus mechanism. Therefore, all role concepts in the consensus mechanism are first introduced.

Client: a client to use the SDK with the multilayer blockchain model for information transmission, then submit trading information, construct good data structure after the delivery trade endorsement node of the operation, the endorsed node is legally requested back to the client, the client obtains a

certain number of endorsers after support can send a legal trade request, and the trading order node is sent to sorting work.

Endorser peer: The endorsement node is responsible for receiving data information sent by the client to apply for the transaction. The node mainly carries out legality detection and ACL simulation transaction permission checks on the transaction through the platform's certificate mechanism. If the verification is passed, the transaction message is signed, and the signed transaction message is returned to the client.

Committer peer: The submission node is responsible for maintaining ledger data, receiving the ordered transaction information sent by the order node, and checking the status of these transactions, generating blocks, and recording them in the ledger.

Order peer: The sorting node is responsible for receiving transaction information from the message queue, sorting the received transaction information, and sending the sorted transaction to the committer node via the message queue for subsequent verification.

There are many peer nodes in a body of the alliance chain, among which the peer node can act as both endorser and committer peer.

The consensus process of the alliance chain is shown in Figure 12, and can be roughly divided into five stages:

Certificate application: The user accesses the multilayer blockchain model system through APP or SDK, and the newly registered user obtains the identity-authentication and transaction certificates through a CA. After obtaining the certificate, the user can publish the transaction information in the model.

Endorsement verification: The client submits the transaction request to the endorsement node, and the endorsement node checks transaction information according to the endorsement mechanism and gives the endorsement result, support or reject. The authentication process of an endorsement node is as follows: First, verifying that the signature of the transaction application submitted by the client is correct. Second, a channel channel ACL check is performed against the identity and transaction certificates to verify that the transaction can take place on the specified channel. Then, the endorsement node performs a simulated transaction to verify whether the transaction conforms to the rules. Finally, the endorsement node endorses the transaction information requested by the verified client, adds the digital signature of the endorsement node, and resends the modified transaction information to the client. A transaction in which a client invokes a smart contract must be endorsed to be considered legal. The transaction may require the unanimous consent of a principal member, or have more than a certain number of individual nodes agree. The deployed chain code on the endorsement node is responsible for details of these policies.

Client-verification stage: The client obtains the transaction information sent by the endorsement node, and first verifies whether the digital signature of the endorsement node is correct. Second, the messages sent by all endorsement nodes are compared. Finally, it checks if the transaction conforms to the endorsement policy. The client sends the transaction that conforms to the endorsement policy to the ordering peer, and also sends transaction information to the endorsement node.

Consensus-sequencing stage: The sequencing node orders peers to only perform the sorting operation and does not read the specific content of the transaction. Generally, the sorting service is provided by the sorting node cluster set up by the authority in the alliance chain. Sort nodes in the cluster are capable of sorting services for different transactions on different channels, which are mainly completed by back-end plug-ins provided by a Kafka message queue. The ordering node orders peers to sort the transactions according to certain rules, processing transactions in units of a period of time or a certain amount of transaction information.

Check sort: After dealing with the orders, peer-sorted trading blocks are sent to the committer node, the node data blocks the deal for the final validation of the work to first verify the structural integrity of transaction data blocks. If the endorsement signature is in line with the endorsement mechanism in the transaction data, it checks whether the current data are trading effectively, and so

on. The chunk of data that are checked by the committer node are written to the ledger, and the node broadcasts synchronous ledger data to the node that participates in the corresponding channel for the transaction.

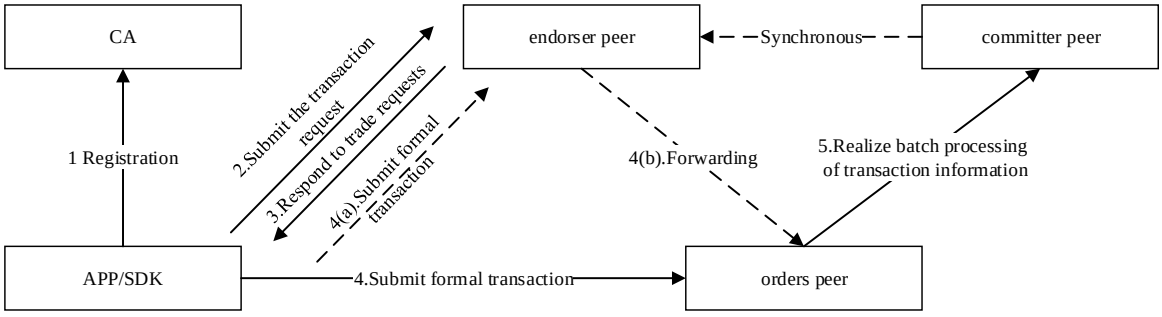

Figure 12. Alliance-chain consensus-mechanism flowchart.

3.7.3. Order-Node Communication-Process Design

A Kafka message queue is a distributed and efficient data queue with its own reliability and fault-tolerant ability to manage data information and ensure consistency between data. An order peer that provides sorting services, uses the enhanced disaster-recovery capabilities of Kafka message queues. When the node machine sends data information to an order peer, if the order-peer server fails and goes down, the data are locally backed up in the Kafka message queue to avoid data loss and other problems. The deployment of order peers and Kafka message queues is shown in Figure 13.

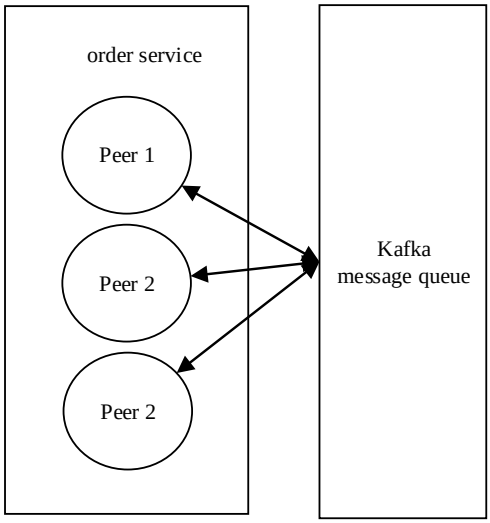

Figure 13. Deployment diagram.

In order to improve the efficiency of transaction sequencing, a cluster of order-peer nodes is generally constructed for processing. This cluster needs to add a Kafka message queue to provide data-transmission and fault-tolerance support. Zookeeper manages Kafka. As shown in the figure above, order peers communicate bidirectionally with Kafka, but the peer does not communicate with another peer. The client can communicate with multiple order peers through the Kafka message queue, which can support the parallel processing of multiple transaction sorts, improve data throughput, reduce transaction delay, and accelerate transaction processing speed.

The BroadCast communication of the Kafka message queue is mainly applied to order peers to obtain transaction data sent by a peer and generate a series of transaction blocks. The peer first sends a message through the gRPC protocol to actively connect with the order peer. After the

connection is completed, the transaction information is sent to the order peer. The order peer receives transaction-data information through the recv function and pushes the information to the Kafka message queue. The consumer side of the Kafka message queue achieves sorts by obtaining the transaction data. The sorted trade data are packaged into a trade block by the CreateNextBlock function and written to the block by the WriteBlock function. The detailed process is shown in Figure 14.

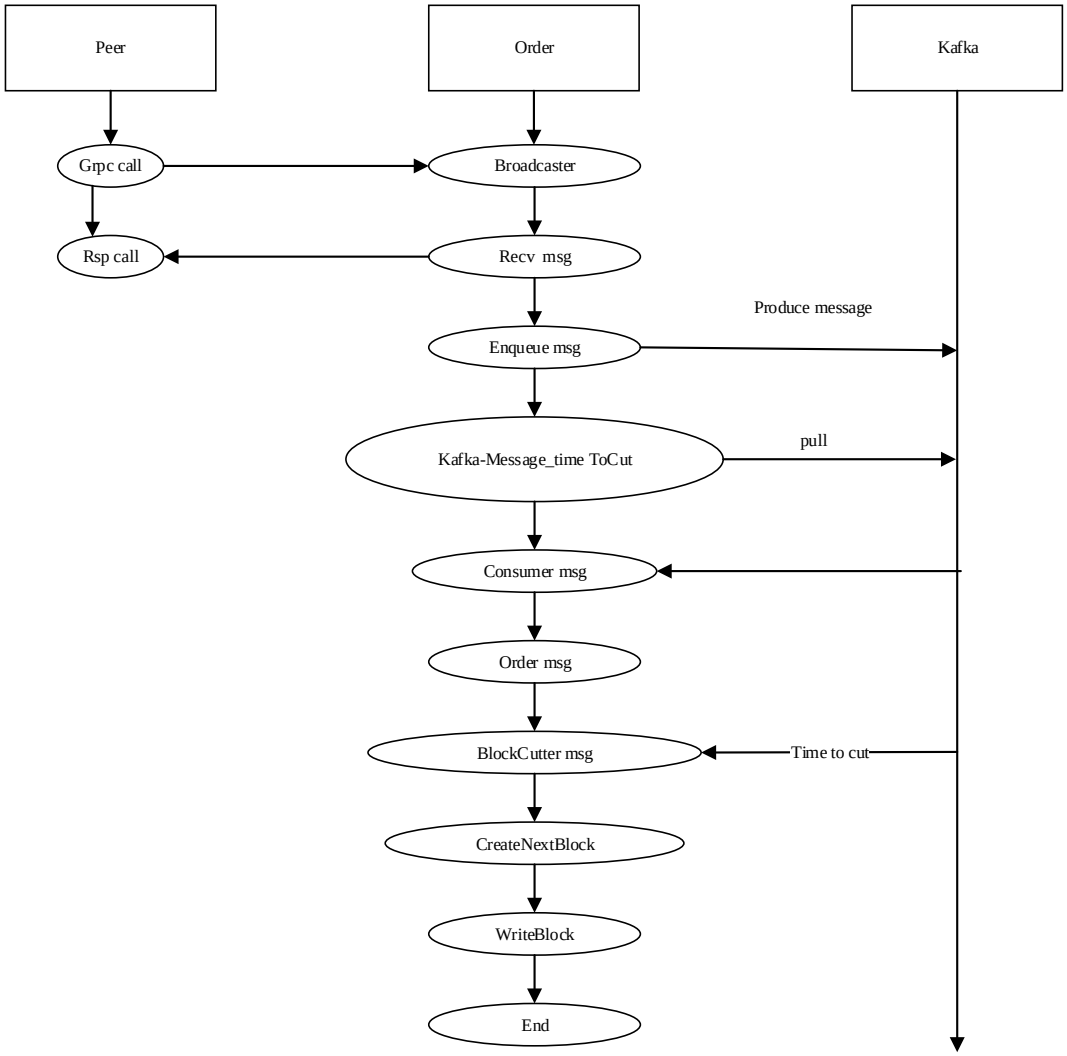

Figure 14. Broadcast schematic.

As shown in Figure 15 below, the delivery method of the Kafka message queue is mainly used for peers to initiate a request from an order peer to obtain a sorted transaction data block. First, the peer initiates a connection with the order peer by using the gRPC protocol, obtaining data through interface function 'get'. The order peer using interface function recv () to obtain the data, in the form of files saved in the latest book data. The order peer calls the SeekInfo interface to obtain a book-data entry, cycle through call iterators get all blocks of book data, returns the block information to the peer, and finally sends the data to the peer to obtain status information.

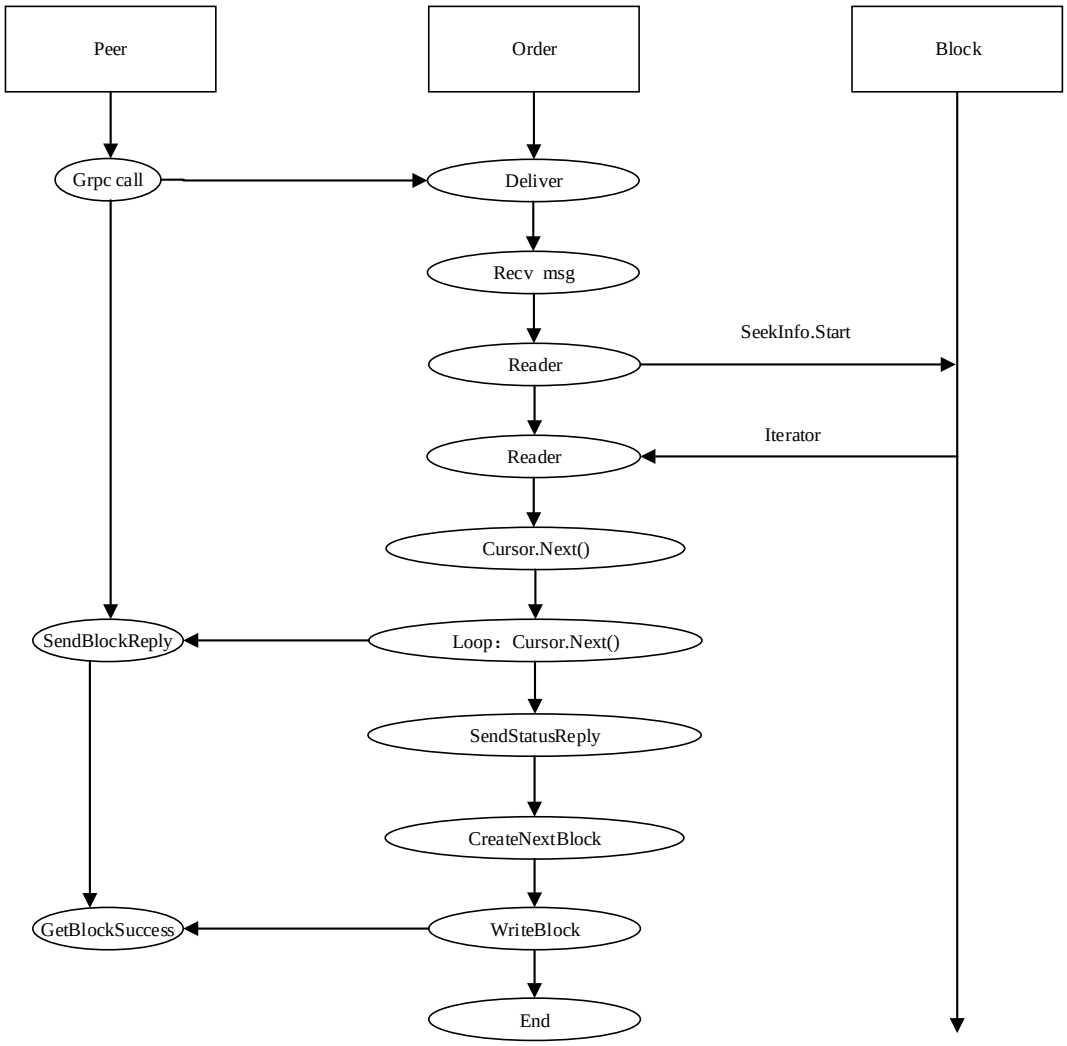

Figure 15. Delivery schematic.

4. Experiment

A study [36] indicated that, in a data-sharing transaction scenario, due to the large amount of data, the original data are usually stored outside the chain, and only the data-core blockchain indicated by the pointer checks the authenticity and accuracy of the data. This paper adopted the same data-storage method as other system models. IoT device data are stored outside the chain, and only references to IoT data are stored on the chain. However, other systems are still in the ideal model stage, with no real implementation, while our system was actually developed. At the same time, no system can be applied to the data-exchange scenario of Internet of things devices.

Our solution is for data-sharing scenarios of IoT devices that currently do not have such systems in the market, so we outline three experiment comparisons on ethereum, a traditional blockchain platform, with regard to privacy access control, bulk data processing, and storage costs and throughput.

4.1. Experiment Environment

The network topology of the blockchain is outlined in Figure 16.

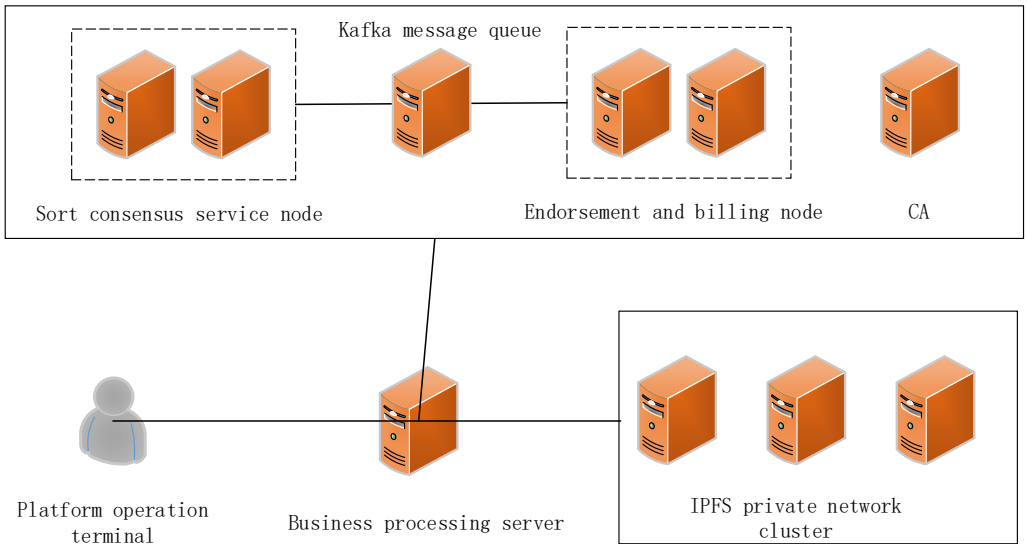

Figure 16. System network topology diagram.

The software required to deploy and install this multicenter blockchain data model is shown in Tables 6 and 7.

Table 6. Software-platform information.

Software	Version	Deployment
Vim	7.4	Every machine
Docker	1.12+	Every machine
Docker Compose	1.8.0+	Every machine
Golang	1.10.1	Every machine
Node	5.6.0	Every machine

Table 7. Deployment platform.

Module	Memory	Location	Numbers
Zookeeper	16G	Docker	6
Kafka	16G	Docker	6
CA	16G	Docker	2
Order peer	16G	Docker	4
Peer	16G	Docker	4
IPFS-System	16G	Storage	3

The blockchain data model proposed in this paper needs to be deployed in a distributed network, and the nodes of different kinds of functions constitute the entire P2P network. The main node types are peer, order peer, Kafka node, and zookeeper node. The peer node is divided into endorser and committer peers. In this platform system, each node runs in a docker container, and nodes communicate by using the gRPC protocol.

4.2. Experiment Program

This experiment compares the blockchain data model designed in this paper with ethereum in terms of privacy access control, data storage, and cost. This experiment was divided into two groups of comparative experiments. The first group of experiments was to create three organizations, org1, org2 and org3. First, systems were tested in terms of privacy access control and massive data-transaction scenarios. Data were simulated by org1 and org2. Then, org1 and org2 performed a data transaction. Simulation user org3 then queried the transaction. The second group of experiments were applied to

our system and to ethereum for five different datasets that were compared in terms of data storage and cost.

4.2.1. Building our Blockchain Platform

First, we needed to create three nodes for org1, org2, and org3. Each organization's CA issued an identity certificate and a transaction certificate to each node, specifying the smart-contract content and platform system version on the chain. Then, nodes join the alliance chain and the private-chain network. Finally, we created Channel1, added org1 and org2 to Channel1, and gave org1 and org2 transaction permissions.

4.2.2. Smart-Contract Run Test

First, we created a client for each of the three nodes. Then, we created a transaction proposal on org1, uploaded the transaction data, broadcasted the transaction proposal, and collected the reply message of the endorsement node. We then input the relevant parameters required in the transaction process, and obtained the serial number ID of the current transaction. After verification was correct, Main-body client org2 executed the smart contract to conduct the transaction; after the transaction was successful, the query method was used to query the transaction-book information. Org1 executed the smart contract and obtained the returned information after the transaction was successful. After the org1 and org2 transactions were completed, transaction information was stored in their respective ledgers for later use. Finally, org3 executed the query method, and queried the transaction data to obtain the result. However, an error was returned as authentication failed.

4.2.3. Experiment Comparison

(a) Experiment 1

Through the above experiment process, the multicenter blockchain model was compared with ethereum in terms of access control and massive-data-storage transactions. Similar to Step 2, in the entire process of simulating transactions in ethereum, three users, user1, user2, and user3, were used for the experiment. The simulated transaction occurred between user1 and user2. After the transaction was successful, the transaction record was stored in the block, and user3 queried the transaction and had the details successfully returned. After the experiment, a comparison between the test results of ethereum and the multicenter blockchain model proved that our system has obvious advantages in terms of privacy access control and massive data transactions.

(b) Experiment 2

In the test process of our blockchain platform, SDK was used to specify the organizational identity relationship of each node in the distributed blockchain. The CA issued the identity and transaction certificates to network nodes, and specified the system version information and the trading channel of the smart contract. Peer1 and peer2, which define two principals, were responsible for generating transactions for their respective clients. First, the client requested a transaction certificate from the CA. Peer1 was responsible for initiating the transaction, and peer2 for receiving the transaction. After the transaction was completed, the experiment data were recorded. The main recorded data were storage space and cost. The above two points were compared with ethereum to prove that the blockchain model introduced in this paper is superior to ethereum in terms of data-sharing transactions.

(c) Experiment 3

In order to better test system performance, we needed to add as many trading users as possible, create 200 new clients to participate in the trading, the data volume of each transaction is 1M, and initiate the trading cycle within a period of time.

Transaction delay defined in this section refers to the time interval between transaction initiation by the client and the final confirmation of the transaction. This parameter measures the data-communication capability and consensus mechanism of the whole blockchain network. A system with lower delay can quickly confirm a transaction, which can reduce client waiting time and enable

users to have better system experience. Throughput is an important parameter that determines transaction delay. Throughput is important reference data that measure a system's acceptance, response to requests, transmission of transaction information, processing of various transaction transactions, and the concurrency of the system in unit time. This parameter is generally expressed as TPS, which represents the total number of transactions generated by the system over a period of 1 second. Formula (2) lists the solution method of this parameter:

$$TPS_t = \frac{Transactions_t}{t}, \tag{2}$$

where $Transactions_t$ represents the total number of transactions that occur in a period of time, t is a period of time, and we set parameter t by ourselves.

4.3. Experiment Program

Table 8 shows the functionality supported by our blockchain system and the three data-storage methods of ethereum as found from Experiment 1.

As shown in Table 8, ethereum cannot be applied to the application scenario of data-transaction access control. We can also see that only the log-event storage method of ethereum can support massive-data transfers that are greater than 1 MB. Therefore, in summary, the blockchain system proposed in this paper has advantages in terms of privacy, access control, and large data-file transactions.

Table 8. Access-control and data-storage function support records.

	Feature Support	Access Control	Access Control
Ethereum platform	Block storage	no	no
	Contract variable	no	no
	Log event storage	no	yes
Multi-layer blockchain model		yes	yes

Table 9. Effect of transaction data size on storage consumption.

	Transaction Data Size	80 B	32 KB	600 KB	1.2 MB	5.4 MB
Ethereum platform	Block storage	80 B	-	-	-	-
	Contract variable	80 B	32 KB	-	-	-
	Log event storage	80 B	32 KB	600 KB	1.2 MB	5.4 MB
Multi-layer blockchain model		32 B	32 B	32 B	32 B	32 B

In Table 9 we show the results from Experiment 2 for storage-space consumption required by the different blockchain platforms for varying sizes of transaction data.

In ethereum, the maximum storage space that the block storage method can support is 80 B, and the fixed fee for each transaction is 21,000 gas. The extra cost for each stored byte is 68 gas, so the total cost of storing 80 B data is 26,440 gas. The contract variable storage method can store up to 32 KB. The transaction fixed cost is the same as before, and variables for store operations cost 20,000 gas. Each 1 KB of data costs 68 gas, so this makes a total of 43,176 gas. The third storage method uses smart-contract log events to store data. This can satisfy all transaction-data sizes for Experiment 2. Each log costs 375 gas, each 1 KB of data stored in the log costs 8 gas. The fourth type, the blockchain system introduced in this paper, only needs to record the 32 B hash value calculated through the IPFS protocol on the file, multiply it by 68 gas, add 21,000 gas, and the calculated gas value is 23,176 gas.

It can be clearly seen from Figure 17 that the data capacity supported by ethereum's Methods 1 and 2 is limited and cannot support massive amounts. Method 3 can support massive-data-sharing transactions, but as the amount of data increases, the cost of required gas linearly increases.

The multicenter blockchain model proposed in this paper adopted the storage method under the file-data chain. Only the hash value of the relevant file data is stored in the chain, which saves storage consumption from the blockchain. At the same time, this blockchain platform is different from public chains such as bitcoin and ethereum. It does not need to spend extra money in the transaction process. Therefore, in summary, the platform compares favorably with existing blockchain technologies such as bitcoin and ethereum.

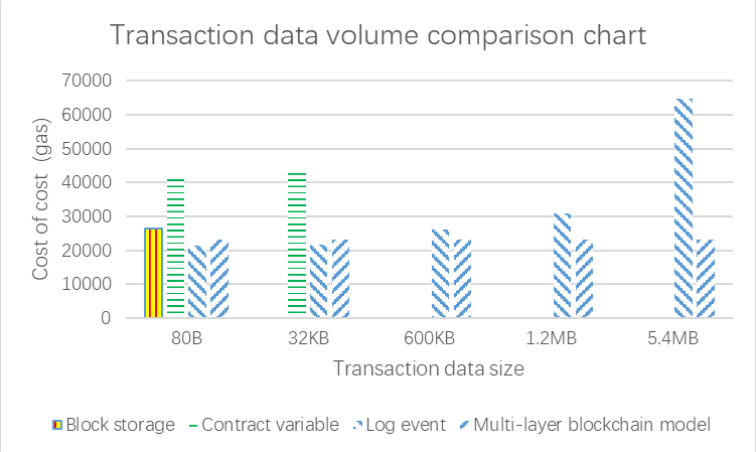

Figure 17. Ethereum transaction-data-volume comparison chart.

Experiment 3 mainly analyzed the TPS parameters. By comparing the *TPS* parameters of bitcoin, ethereum, and our platform, the performance of the consensus algorithm introduced in this chapter was compared. According to Formula 1, we took time interval t as 10, 20, 40, 80, 160, and 320 s, respectively, and tested each time interval tenfold. The data of each test are shown in Table 10.

According to the recorded data condition in Table 10, each interval separately averaging 10 data, we concluded that each interval of the average number of trading, through six obtained *TPS* value formulas, based on the conclusion from six *TPS* average blockchain systems in the *TPS* experiment was about 1500 or so.

Table 10. Total transaction record.

The Time Interval	1	2	3	4	5	6	7	8	9	10
10s	1.42	1.46	1.48	1.92	1.39	1.67	1.28	1.36	1.90	1.34
20s	3.84	2.78	3.34	2.56	2.72	3.8	2.68	2.84	2.92	2.96
40s	4.26	4.38	4.44	5.76	4.17	5.01	3.84	4.08	5.7	4.02
80s	5.36	5.68	5.84	5.92	7.68	5.56	6.68	5.12	5.44	7.6
160s	7.1	7.3	7.4	9.6	6.95	8.35	6.4	6.8	9.5	6.7
320s	7.68	8.16	11.4	8.04	8.52	8.76	8.88	11.52	8.34	10.0

As shown in Figure 18, compared with current mainstream blockchain-technology platforms bitcoin and ethereum, the method proposed in this paper has obvious advantages in terms of parameter TPS. Currently, mainstream blockchain platforms bitcoin and ethereum have relatively low TPS throughput parameters mainly because a high-throughput approach is prone to fork problems. However, there is no bifurcation problem in the construction of an alliance chain. Therefore, the consistency algorithm proposed in this paper can be well-applied to the multilayer blockchain model proposed in this paper to achieve high throughput, reduce latency, and improve transaction processing speed.

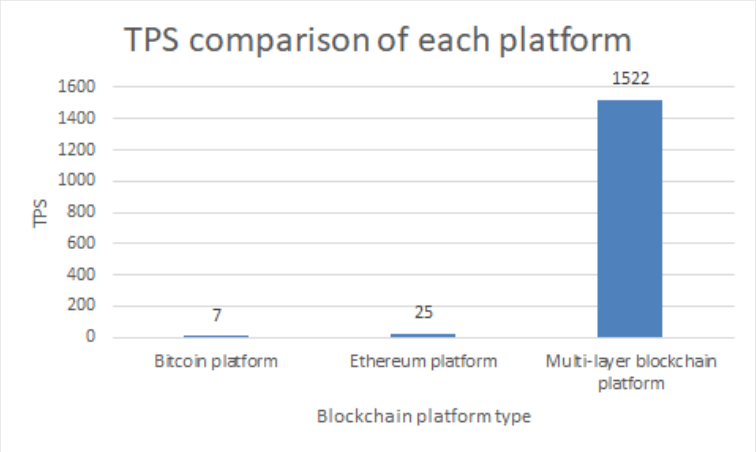

Figure 18. Ethereum transaction-data-volume comparison chart.

5. Summary

Faced with the privacy and security of mass data exchange and transmission in the current Internet of Things environment, this paper proposed a layered authorization transaction multilayer blockchain model based on mass data support: Data-model design, structure design, business-intelligence contract-authority management and identity-authentication-mechanism design, access-control-mechanism design, data storage, etc. Different from the single-chain model of bitcoin and ethereum, this paper proposed a double-chain model in which the main alliance chain is responsible for processing transactions, and the private chain is responsible for storing transaction data. The alliance-chain network is deployed between each subject, and the private-chain network is deployed within each subject. Through the introduction of authority management and the identity-authentication mechanism, the identity and transaction certificates are issued to each node in the alliance chain, and the access-control function in the transaction process is realized through the channel mechanism. The blockchain platform uses chain storage to store real data in the IPFS cluster server built by the alliance. The index hash of the blockchain storage file greatly reduces consumption of blockchain storage space. Through comparative experiments, we found that the data models of bitcoin and ethereum have advantages in data storage and access control. In addition, our consensus algorithm that fits the multilayer blockchain model can better adapt to a scenario where IoT devices generate a large amount of data, and ensure that data generated by IoT devices have good privacy and security, as well as high throughput and efficiency in the process of exchange and sharing.

Finally, there are two future works to be solved by the authors and other scholars. The first is the design of an efficient incentive algorithm. This paper did not design an efficient incentive algorithm to reward those who are willing to actively share data. In addition, the consensus algorithm used in this paper can be further designed to improve the speed of transaction processing.

Author Contributions: Conceptualization, S.X., Y.Z. and W.Y.; methodology, Y.Z., W.W. and I.C.; software, Y.Z.; validation, S.X. and W.Y.; data curation, H.T.; writing—original-draft preparation, Y.Z. and H.T.; writing—review and editing, S.X. and H.T.; supervision, S.X. and I.C.; project administration, W.Y.

Funding: This work was supported by the National Natural Science Foundation of China (61802086) and the Fundamental Research Funds for the Central Universities (3072019CFM0601, 3072019CF0603).

Acknowledgments: This work has been supported by the China Scholarship Council (CSC).

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

References

1. Pronk, T.E.; Wiersma, P.H.; van Weerden, A. A Research data Sharing Game. *PeerJ PrePrints* **2014**, *2*, e599v1. [CrossRef]
2. Sinaci, A.A.; Erturkmen, G.B.L. A federated semantic metadata registry framework for enabling interoperability across clinical research and care domains. *J. Biomed. Inform.* **2013**, *46*, 784–794. [CrossRef] [PubMed]
3. Dennis, R.; Owen, G. Rep on the block: A next generation reputation system based on the blockchain. In Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), London, UK, 14–16 December 2015, pp. 131–138. [CrossRef]
4. Bitcoin Wiki. Scalability. Available online: https://en.bitcoin.it/wiki/Scalability (accessed on 15 February 2019).
5. Narayanan, A.; Bonneau, J.; Felten, E.; Miller, A.; Goldfeder, S. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*; Princeton University Press: Princeton, NJ, USA, 2016.
6. Decker, C.; Seidel, J.; Wattenhofer, R. Bitcoin meets strong consistency.In Proceedings of the 17th International Conference on Distributed Computing and Networking, Singapore, 4–7 January 2016; pp. 1–10. [CrossRef]
7. Anish Dev, J. Bitcoin mining acceleration and performance quantification. In Proceedings of the 2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE), Toronto, ON, Canada, 4–7 May 2014; pp. 1–6. [CrossRef]
8. Guadamuz, A.; Marsden, C. Blockchains and Bitcoin: Regulatory responses to cryptocurrencies. *First Monday* **2015**, *20*, 12. [CrossRef]
9. Tschorsch, F.; Scheuermann, B. Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2084–2123. [CrossRef]
10. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. Available online: https://bitcoin.org/en/bitcoin-paper (accessed on 28 November 2019).
11. Wood, D.G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32.
12. Gensichen, J.; Vollmar, H.C.; Sönnichsen, A.; Waldmann, U.M.; Sandars, J. E-learning for education in primary healthcare—Turning the hype into reality: A Delphi study. *Eur. J. Gen. Pract.* **2009**, *15*, 11–14. [CrossRef] [PubMed]
13. González, G.R.; Organero, M.M.; Kloos, C.D. Early Infrastructure of an Internet of Things in Spaces for Learning. In Proceedings of the 2008 Eighth IEEE International Conference on Advanced Learning Technologies, Santander, Spain, 1–5 July 2008; pp. 381–383. [CrossRef]
14. Sarma, A.C.; Girão, J. Identities in the Future Internet of Things. *Wirel. Pers. Commun.* **2009**, *49*, 353–363. [CrossRef]
15. Lee, S.; Bae, M.; Kim, H. Future of IoT Networks: A Survey. *Appl. Sci.* **2017**, *7*, 1072. [CrossRef]
16. Kiyomoto, S.; Rahman, M.S.; Basu, A. On blockchain-based anonymized dataset distribution platform. In Proceedings of the 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA), London, UK, 7–9 June 2017; pp. 85–92. [CrossRef]
17. Chen, J.; Xue, Y. Bootstrapping a Blockchain Based Ecosystem for Big Data Exchange. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 460–463. [CrossRef]
18. Yue, X.; Wang, H.; Jin, D.; Li, M.; Jiang, W. Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control. *J. Med Syst.* **2016**, *40*, 218. [CrossRef] [PubMed]
19. Peterson, K.; Deeduvanu, R.; Kanjamala, P.; Boles, K. A Blockchain-Based Approach to Health Information Exchange Networks. In Proceedings of the NIST Workshop Blockchain Healthcare, Gaithersburg, MD, USA, 26–27 September 2016; pp. 1–10.
20. Zhang, J.; Xue, N.; Huang, X. A Secure System For Pervasive Social Network-Based Healthcare. *IEEE Access* **2016**, *4*, 9239–9250. [CrossRef]
21. Moran, T.P.; Carroll, J.M. *Design Rationale: Concepts, Techniques, and Use*; L. Erlbaum Associates Inc.: Hillsdale, NJ, USA, 1996.
22. Zamfir, V. Introducing Casper "the Friendly Ghost". Available online: https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost (accessed on 1 August 2015).

23. Atzei, N.; Bartoletti, M.; Cimoli, T. A Survey of Attacks on Ethereum Smart Contracts (SoK). In *Principles of Security and Trust*; Maffei, M.; Ryan, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; pp. 164–186.
24. Corbet, S.; Lucey, B.; Yarovaya, L. Datestamping the Bitcoin and Ethereum Bubbles. *Financ. Res. Lett.* **2018**, *26*, 81–88. [CrossRef]
25. Buterin, V. Ethereum Scalability Research and Development Subsidy Programs. 2018. Available online: https://blog.ethereum.org/2018/01/02/ethereum-scalability-research-development-subsidy-programs/ (accessed on 28 November 2019).
26. Buterin, V. A Next Generation Smart Contract & Decentralized Application Platform. 2014. Available online: https://www.ethereum.org/ (accessed on 28 November 2019).
27. Gaur, N.; Desrosiers, L.; Novotny, P.; Ramakrishna, V.; O'Dowd, A.; Baset, S. *Hands-On Blockchain with Hyperledger: Building Decentralized Applications with Hyperledger Fabric and Composer*; Packt Publishing Ltd: Birmingham, UK, 2018.
28. Kypriotaki, K.; Zamani, E.; Giaglis, G. From Bitcoin to Decentralized Autonomous Corporations—Extending the Application Scope of Decentralized Peer-to-Peer Networks and Blockchains. In Proceedings of the 17th International Conference on Enterprise Information Systems. SCITEPRESS—Science and and Technology Publications, Barcelona, Spain, 27–30 April 2015; pp. 284–290. [CrossRef]
29. Eyal, I.; Gencer, A.E.; Sirer, E.G.; van Renesse, R. Bitcoin-NG: A Scalable Blockchain Protocol. Available online: https://arxiv.org/abs/1510.02037v1 (accessed on 28 November 2019).
30. Snow, P.; Deery, B.; Lu, J.; Johnston, D.; Kirby, P. Factom White Paper. Available online: https://www.factom.com/factom-blockchain/ (accessed on 28 November 2019).
31. Kung, H.T.; Robinson, J.T. On Optimistic Methods for Concurrency Control. *ACM Trans. Database Syst. (TODS)* **1981**, *6*, 213–226. [CrossRef]
32. Cachin, C. Architecture of the Hyperledger Blockchain Fabric. In Proceedings of the Workshop on Distributed Cryptocurrencies and Consensus Ledgers, Chicago, IL, USA, 25 July 2016; Volume 310, p. 4.
33. Chen, Y.; Li, H.; Li, K.; Zhang, J. An improved P2P file system scheme based on IPFS and Blockchain. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; pp. 2652–2657. [CrossRef]
34. Yao, A. How to generate and exchange secrets. In Proceedings of the 27th Annual Symposium on Foundations of Computer Science (sfcs 1986), Toronto, ON, Canada, 27–29 October 1986; Volume 10, pp. 162–167. [CrossRef]
35. Sousa, J.; Bessani, A.; Vukolic, M. A Byzantine Fault-Tolerant Ordering Service for the Hyperledger Fabric Blockchain Platform. In Proceedings of the 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Luxembourg, 25–28 June 2018; p. 11.
36. Xie, J.; Tang, H.; Huang, T.; Yu, F.R.; Xie, R.; Liu, J.; Liu, Y. A Survey of Blockchain Technology Applied to Smart Cities: Research Issues and Challenges. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2794–2830. [CrossRef]