Networking and Data Storage Mechanism in Blockchain

Xiangwei Zheng*

College of Arts and Sciences, Case Western Reserve University, Cleveland, US, 44106

*Corresponding author: applicantMagie@163.com

Abstract. This paper proposes a blockchain networking scheme based on cloud platform deployment and its adapted data. Firstly, a subnet-based non-all-connection networking scheme is formed, which limits the scope of transaction confirmation to a limited number of nodes. Secondly, the data sharing mechanism of cloud computing divides the data into three levels: transaction data-sensitive state data-non-sensitive state data. The node only saves the transaction data related to the state transition to ensure the non-destructive modification, and the state data is supported by the cloud. The calculated features enable different levels of shared storage, maximizing storage space. The experimental results show that the scheme can provide new ideas for the storage and sharing of trusted data in the blockchain.

Keywords: Cloud Computing, Blockchain, Data Sharing.

1. Introduction

1.1. Motivation

In recent years, blockchain has become a hot topic in society. It integrates technologies such as peer-to-peer transmission, distributed data storage, consensus mechanisms, encryption algorithms, and has characteristics such as tamper resistance, traceability, and decentralization. These characteristics make blockchain very suitable for storing and sharing trusted data.

Currently, research on the storage and sharing mechanism of blockchain data mainly faces two issues. One is the efficiency of transaction confirmation. Due to the fact that each update of blockchain status requires confirmation from a majority of nodes across the network, forming a consensus, and broadcasting to each node to achieve synchronization of the ledger across the network, the more nodes participate in the network, the slower the transaction confirmation speed. The second issue is the utilization of storage space. Due to the immutable and non removable nature of blockchain, the data stored on the blockchain will only continue to grow over time, resulting in significant long-term storage space overhead.

1.2. The related work

By introducing concepts related to cloud computing, the above issues can be improved to a certain extent. The combination of blockchain and cloud computing, known as BaaS (Blockchain as a Service), can effectively reduce the cost of blockchain deployment. A blockchain service quickly built using cloud computing. In November 2015, Microsoft provided BaaS service on the Azure [2] cloud platform and officially opened it to the public in August 2016. Developers can create blockchain environments on it in the simplest and most efficient way. IBM [3] also announced the launch of the blockchain service platform in February 2016, helping developers create, deploy, run, and monitor blockchain applications on the IBM cloud. Domestic companies such as Alibaba Cloud and Tencent Cloud have also provided BaaS services [4,5].

1.3. Our work

This article proposes a blockchain networking solution based on cloud platform deployment and an adaptive data sharing and storage solution, which mainly has three characteristics.

(1) Introducing the concept of non fully connected networking, limiting the scope of transaction confirmation to a limited number of nodes.

(2) Hierarchical storage of data based on sensitivity improves storage space utilization while preserving immutability.

(3) Design for deployment in cloud computing environments, allowing users to quickly build blockchain services through cloud platforms and reduce deployment costs.

The previous work has improved transaction confirmation efficiency and storage space utilization to varying degrees, but there are also certain limitations. Table 1 compares and evaluates the proposed approach from four perspectives with the aforementioned research results.

Table 1. Comparison

	Connection	Cloud Environment	Efficiency	utilization rate
[6]	Fully connected	No	Middle	Middle
[7]	Fully connected	No	Low	Low
[8]	Fully connected	No	Middle	Middle
[9]	Fully connected	Yes	Low	Middle
[10]	Fully connected	No	Middle	Middle
[11]	Fully connected	Yes	High	High
This paper	Not Fully connected	Yes	High	High

2. Design of blockchain networking based on cloud computing

2.1. A Non Fully Connected Blockchain Networking

All algorithmic models simulate the behavior of nature or human society, and few individuals in both nature and human society interact with each other, meaning that fully connected models are almost non-existent. Usually, each individual only has close contact with several individuals, which means that the actual model is a set composed of several sparse graphs. Figure 1 shows the actual topology structure of a distributed network.

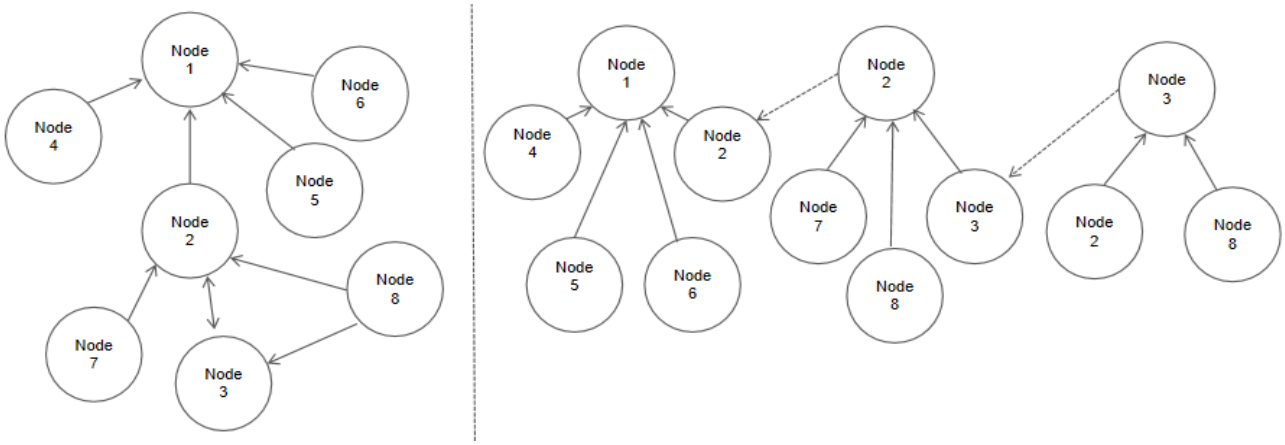

Figure 1. Actual topology structure

In actual network topology, there is no connection between graphs and there is no data interaction between them. Therefore, there is no need to store internal data from other graphs between different graphs, which is the first step in achieving consensus at the network structure level.

Within each graph, the data interaction relationships between nodes can be summarized into two basic models, namely a single master-slave relationship and a mutual master-slave relationship. The left half of Figure 2 shows these two relationships, where the node pointed by the arrow is called the "parent node", and the node emitted by the arrow is called the "child node".

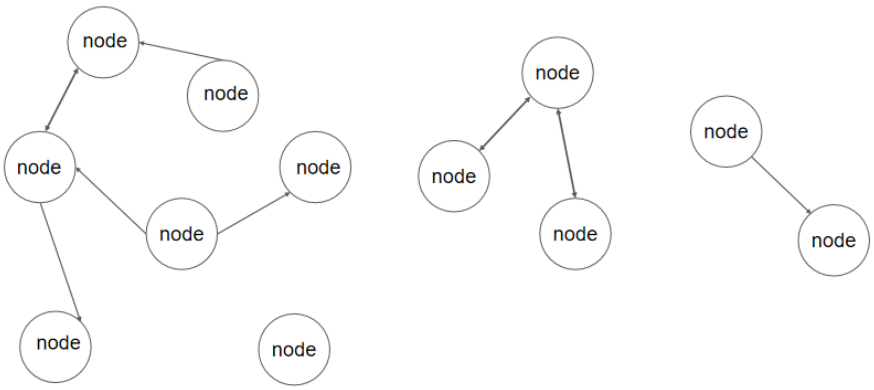

Figure 2. The relationship between adjacent nodes

In the single master-slave relationship model, each "write" request to the ledger is only issued from the parent node, and the child node is only responsible for verifying the request from the parent node. Of course, whether or not the ledger can be legally written ultimately depends on the joint decision of the parent and child nodes. The supply chain is a typical structure dominated by a single master-slave relationship, with data flowing in a one-way manner. For example, transactions can only be initiated by suppliers and confirmed by customers.

In the mutual master-slave relationship model, each node can randomly initiate write requests without obvious membership relationships. WeChat and other instant messaging tools belong to a typical master-slave relationship model, where both parties can "add and write" to their account books (chat records) at any time. Strictly speaking, the mutual master-slave relationship model can also be decomposed into two independent single master-slave relationship models. Whether further segmentation is necessary can be determined based on the size of the network and the degree of correlation between transactions between nodes. Generally speaking, the larger the network size and the lower the relevance of transactions, the more subdivision is needed.

We first truncate the long path of the original graph and classify all edges pointing to the same node into the same class, forming a basic single-layer structure. We refer to it as an "atomic tree", which means a tree that is no longer subdivided. A feasible approach is to scan the original image in sequence, identify all parent nodes (i.e., nodes pointed by several arrows), and divide the original image based on each parent node. Then identify the child nodes of each parent node in sequence, and ultimately form several atomic trees. Figure 4 shows the decomposition process, where nodes 1, 2, and 3 are the parent nodes of the atomic tree, and each subtree is connected through "messenger nodes", namely nodes 2 and 3. Obviously, nodes that can become messenger nodes must be adjacent to at least one "in edge" and "out edge" simultaneously.

2.2. Networking Scheme Design

Each node in each atomic tree jointly maintains the data flow within the tree, which is highly correlated. Therefore, we write all the interactive data within the atomic tree in the same distributed ledger. The nodes within the atomic tree together form a blockchain network, which we refer to as a subnet.

Each subnet needs to include the following basic roles. Authorization authentication node: CA, mainly uses digital certificate mechanism to manage member identities in the network. Ordinary node: initiate transactions or sign and endorse transactions. Sorting node: globally sort the received legitimate transactions in the network and package them into blocks. Full node: Verify the legality of blocks and maintain the entire ledger. In addition, authorized external nodes or clients can obtain access to some or all of the data in the ledger.

From the above, we can see that for each new subnet, it is necessary to allocate all nodes and sort nodes. However, in a cloud computing environment, we can flexibly allocate virtual machine resources and achieve automated deployment through scripts. Therefore, this networking solution can

achieve the highest operational efficiency in a cloud computing environment. But it should be noted that in non cloud computing environments, networking solutions are still feasible.

Figure 3 shows the complete data interaction process within an atomic tree with 4 nodes (1 parent node, 3 child nodes). The thicker arrows indicate the information exchange between different modules, while the thinner arrows indicate the information transmission process within the module. The solid line indicates that the interaction process is primary, while the dashed line indicates that the interaction process is secondary.

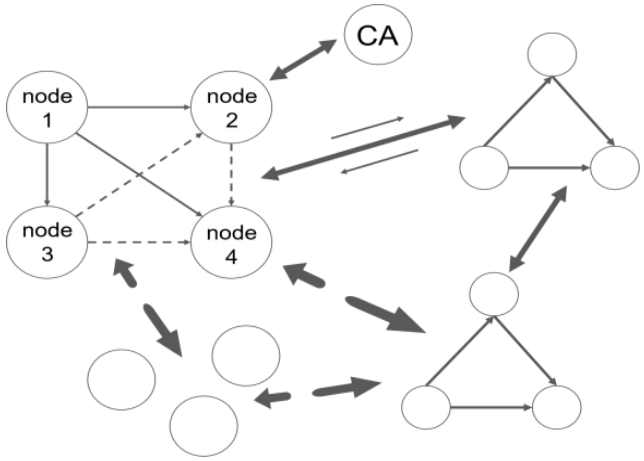

Figure 3. Subnet Architecture and Data Interaction

3. Blockchain Data Storage Mechanism Based on Cloud Computing

3.1. Data storage architecture design

The idea of data storage architecture in this article comes from the Hyperledger Fabric [12] project, which is a representative alliance chain project led by IBM. In fabric, a ledger is a series of orderly and tamper proof state transfer records logs. State transfer is the result of chain code execution (transactions), where each transaction submits a series of key value pairs to the ledger through addition, deletion, and modification operations [13]. A series of orderly transactions are packaged into blocks, connecting the ledger into a blockchain. Meanwhile, a state database maintains the current state of the ledger, hence it is also known as the world state. The ledger status database actually stores the latest values of all key value pairs that have appeared in transactions. Calling chain code to execute transactions can change state data. In order to efficiently execute chain code calls, the latest values of all data are stored in the state database. Logically speaking, a state database is only a snapshot of an ordered transaction log, so it can be regenerated at any time based on the transaction log.

Based on this idea and combined with the characteristics of cloud computing, this article has expanded to form a blockchain data storage mechanism based on cloud computing. This mechanism divides the data that needs to be stored into two parts: transaction data and state data. Transaction data contains key content related to state transitions, with the aim of ensuring the order preservation and tamper resistance of the entire network. It is stored through a blockchain structure, and a block should contain the following fields.

Transaction Details: Indicates the actual transactions that occur between the relevant nodes. The generator identifier of the transaction: indicates which node generated the transaction, that is, the complete signature of the generator. Verifier ID of the transaction: The recipient of the transaction confirms that the transaction details and its generator ID are legal, and performs the final signature on the transaction. Unique identifier of the previous block: indicates which block the current block data is appended to, and this identifier is usually the hash value serialized from the previous block. Block number: indicates which block the current block is globally. Unique identifier of the current block: hash value after serialization of the current block. Block type: Indicates the type of current block, such as regular block or Genesis block.

The state data is similar to the concept of world state mentioned earlier, and represents the global state of the entire subnet. There are two options available in fabric: LevelDB or CouchDB, but these two databases can support fewer query methods and cannot meet the complex query needs of commercial applications well. In fact, we believe that as long as transaction data can be used to ensure the immutability of the ledger, status data can be stored in other high-performance databases.

Considering that in practical applications, users may have different requirements for the encryption level of data, the state data can be divided into sensitive data that is only visible to internal members of the subnet, and non sensitive data that can be shared with the outside world.

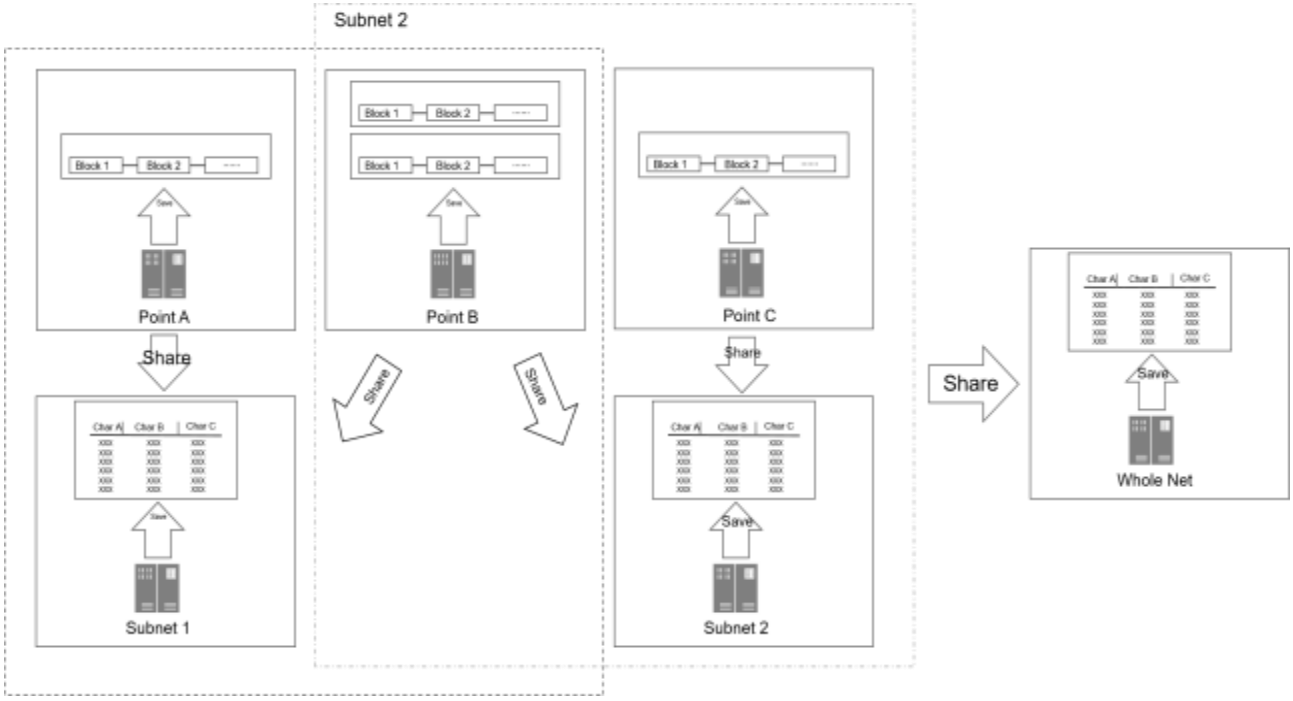

Figure 4. Blockchain Data Storage Architecture

Taking Figure 4 as an example, node a and node b form subnet 1, while node b and node c form subnet 2. Node B stores transaction data for both subnets simultaneously. The two subnets are equipped with a full node cluster for storing sensitive data in the subnet, which is shared by nodes within the subnet. For non sensitive data from each subnet, it can be stored in the entire network data sharing server cluster.

The state data in this article's solution is stored through the HBase database deployed on the cloud server. Hbase [14] is a distributed, column oriented open source database. This technology is derived from the BigTable [15] paper written by Fay Chang and is a high-performance storage solution for massive unstructured data. HBase supports compression storage function based on Snappy [16] algorithm, which can save storage space to the maximum extent. Meanwhile, due to the ownership of transaction data directly related to state change records, nodes can still effectively prevent the tampering of state data.

3.2. Typical Process Analysis

3.2.1 Subnet formation process

1.All types of nodes need to apply for corresponding certificates from the CA before joining the network.

2.When a node wants to create a subnet, it first obtains the corresponding member's certificate, IP, permissions, etc. from the CA, and initiates networking requests to other nodes based on the IP.

3.After the networking rules (such as all nodes agreeing) are passed, the node that initiated the initial request generates a signed Genesis block, and assigns sorting nodes and all nodes to the subnet.

3.2.2 Data Storage Process

1.The prerequisite for any node in the subnet to generate a transaction/record is that the corresponding certificate has been applied to the CA to prove its identity.

2.Transactions initiated by nodes within a subnet require the corresponding nodes to sign the transactions based on the number of members involved in the subnet. For example, if node A initiates a transaction involving A, B, and C, the transaction must contain all signatures of A, B, and C to be legal. At this time, offline transactions cannot be conducted. If the transaction only involves A itself, the transaction can be conducted offline.

3.After collecting all signatures, node A sends the transaction with a sequence number (indicating the current global transaction) to the sorting node, which uses consensus/fault tolerance algorithms to return an accept or reject response.

4.If the sorting node accepts the transaction, it will be signed and broadcasted to all nodes along with the hash of the transaction content, which is equivalent to synchronously adding a new transaction data to all nodes.

5.At the same time as the previous step, the sorting server will divide the transaction into two types: sensitive data and non sensitive data. Sensitive data will be submitted to the entire node server of the subnet for storage, while non sensitive data will be submitted to the shared server of the entire network for storage. It is shown in Figure 5.

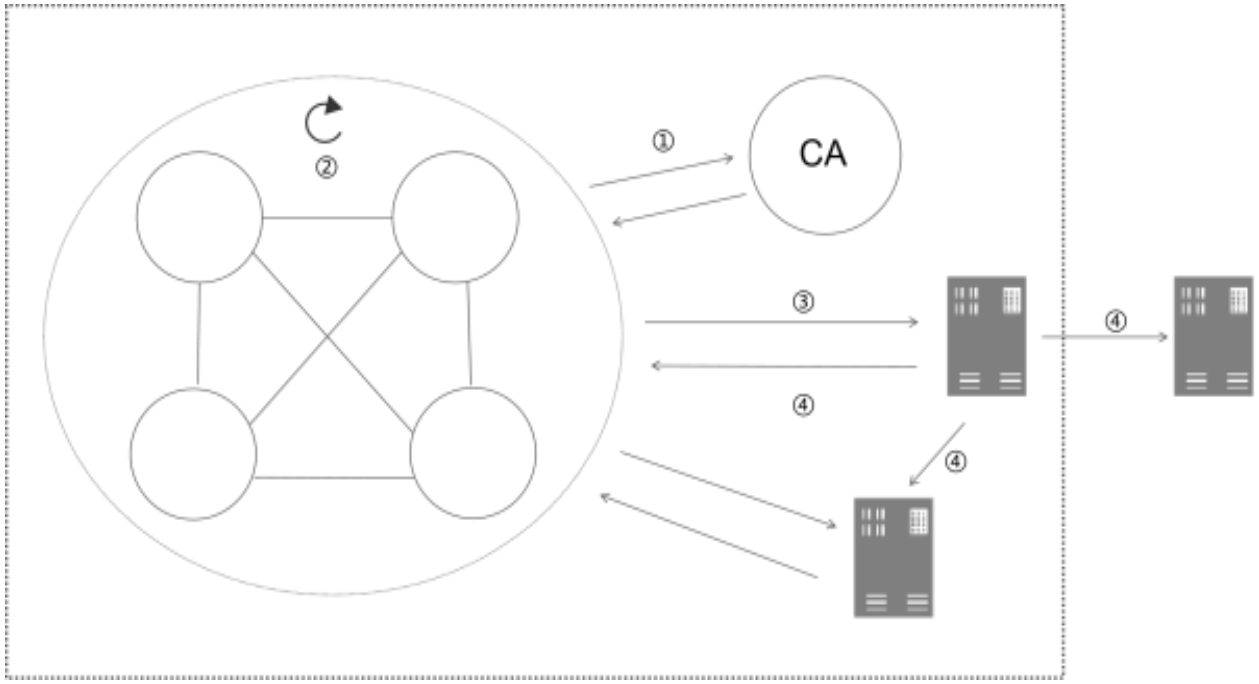

Figure 5. Data Storage Process

3.2.3 Data query process

From the comparison between prediction data and actual data, the BP neural network has better prediction performance and relatively small error, which can meet the demand completely, and has fast prediction speed and convenient operation. Figure 6 shows the specific details.

1.The node initiates a query request through the client, and the node to which the node belongs first verifies the client's identity and permissions.

2.If the verification is successful, initiate queries to the entire network data sharing server and the entire subnet node server respectively.

3.The two types of services in the previous step respectively return the requested sensitive and non sensitive data field information. After combining them, the node server takes a hash value and verifies it with its stored transaction data.

4.Node returns data to the client.

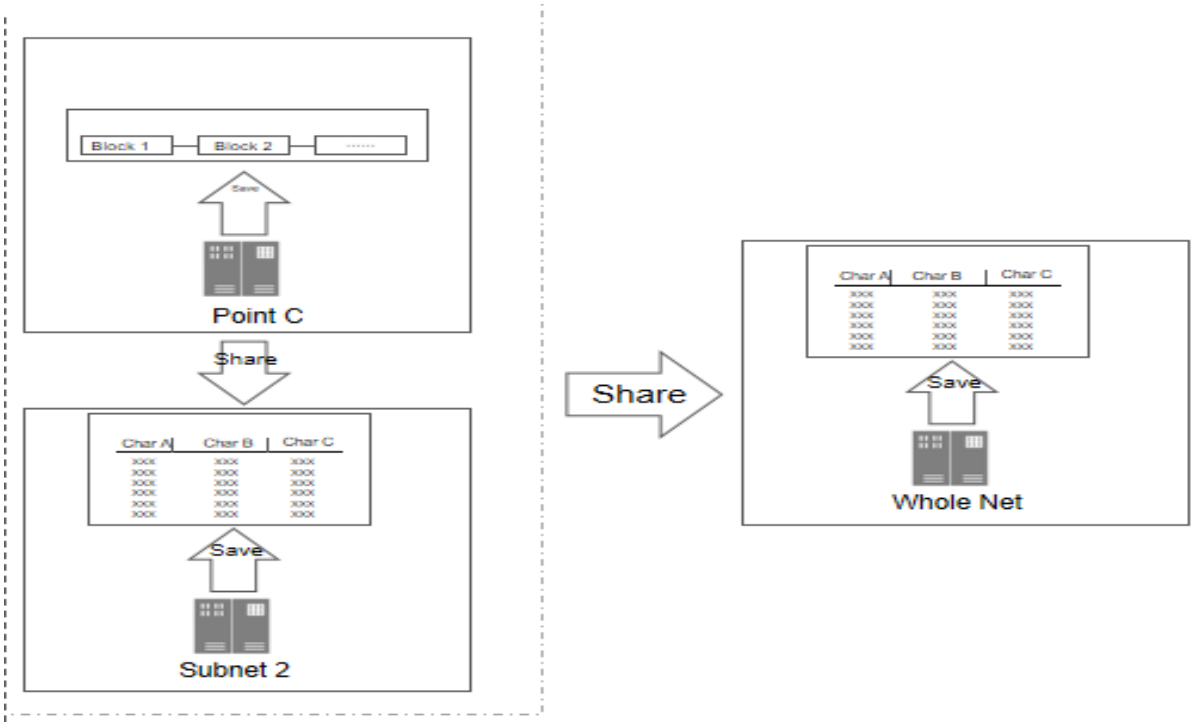

Figure 6. Data Query Process

4. Experiment and analysis

Supply chain is currently a popular application scenario for blockchain technology, and it is also a typical structure with a single master-slave relationship as the main focus. In the supply chain, data is basically unidirectional, with transactions shipped by suppliers and confirmed by customers.

In the basic data model of the supply chain, each upstream node can be regarded as the parent node of a subnet, and its downstream is the child node. At the same time, if the downstream itself conducts transactions with further downstream, it can also become the parent node. By using a non fully connected model, a large supply chain can be divided into a subnet, which will be tested and analyzed based on this model in the following text.

In the traceability application of agricultural product supply chain mentioned earlier, the data is mainly stored in the following 5 tables.

Product: The main product information of the current node, including product name, place of origin, price, etc. This data is encrypted and stored in the entire node server of the subnet.

Customer: customer information of the current node, including customer name, contact number, ID number, business license, etc. This part of data is encrypted and stored in all node servers of the subnet.

Supplier: The supplier information of the current node, including supplier name, contact phone number, business license, business scope, address, etc. This data is encrypted and stored in the entire node server of the subnet.

Purchase: The purchase record of the current node, including purchase order number, transaction date, purchase details (including product, batch, quantity, price, etc.), supplier, total price, transportation logistics document, etc. The date, product, quantity, etc. are not sensitive data without exposing the node identity, so they can be stored on the network wide data sharing server, while the remaining information is encrypted and stored on the subnet wide node server.

Sale: The sales record of the current node, including sales order number, transaction date, sales details (including product, batch, quantity, price, etc.), customer, total price, transportation logistics document, etc. The date, product, quantity, etc. are not sensitive data without exposing the node identity, so they can be stored on the network wide data sharing server, while the rest of the information is encrypted and stored on the subnet wide node server.

4.1. Results

Split 100 nodes in the basic data model of the supply chain and test the throughput of fully connected and non fully connected networking schemes under different node numbers.

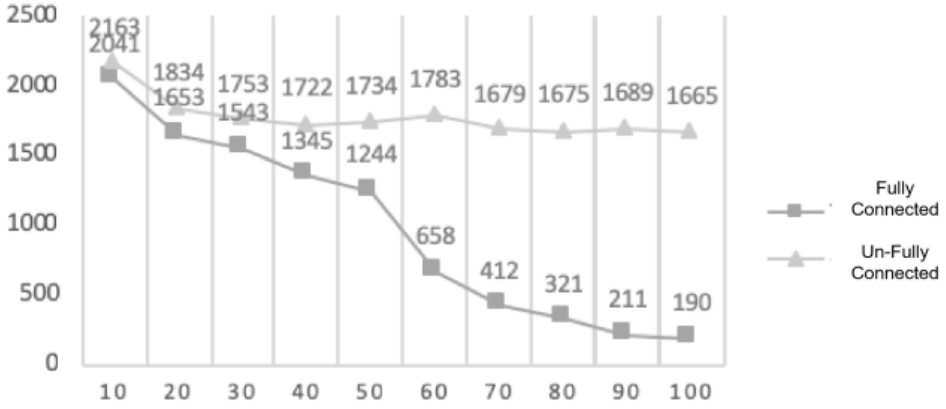

Figure 7. Throughput Comparison

As shown in Figure 7, the fully connected networking scheme has higher throughput when the number of nodes is small. However, as the number of nodes increases, the throughput will significantly decrease. This is because in the fully connected model, each corresponding node updates a state and needs to broadcast a network request to all other nodes. The total network request volume is N * (M-1), where M is the total number of nodes in the distributed network, so the more nodes there are, The longer the transaction confirmation time, the lower the throughput. However, the non fully connected networking scheme splits a large fully connected network into different subnets, and transactions only require confirmation from relevant parties rather than from all nodes. Therefore, as the number of nodes increases, the trend of throughput decline is not significant. At 100 nodes, the throughput of non fully connected networking reaches 8.8 times that of fully connected networking.

4.2. Comparison of storage space efficiency

This experiment consists of 10 nodes randomly generating 10 million transaction data.

If we follow the classic blockchain data storage scheme, where each node saves a complete ledger, the total storage space occupied is approximately 331GB.

Using the data sharing storage mechanism proposed in this article, the data is managed by dividing it into three levels: transaction data, sensitive state data, and non sensitive state data. And use the HBase database deployed on the cloud server for compressed storage, occupying a total storage space of 107GB, saving up to 67% of storage space.

5. Conclusion

This paper optimizes the deployment of blockchain in cloud computing environments and proposes a new blockchain networking mechanism to improve the overall throughput of the network; At the same time, by introducing cloud computing based data sharing storage, the storage space has been maximized. Further in-depth research will be conducted on the theoretical basis of this article, including the comparison of storage space, query latency, and throughput of different data sharing models applied in blockchain data storage; The risk resistance of the model in various unexpected situations (node offline, malicious node attacks, forking, etc.

References

[1] Samaniego M, Deters R. Blockchain as a Service for IoT [C]. 2016 IEEE International Conference on Internet of Things. 2016: 433-436.

[2] Microsoft. Azure Blockchain Solution Architecture [EB/OL]. 2019-7-15. https://azure.microsoft.com/en-us/solutions/blockchain.

[3] IBM. The IBM Blockchain Platform [EB/OL]. 2019-7-15. https://cloud.ibm.com/docs/services/blockchain?topic=blockchain-ibm-blockchain-platform#get-started-ibp.

[4] Alibaba. Alicloud blockchain service [EB/OL]. 2019-7-15. https://cn.aliyun.com/product/baas.

[5] Tencent. Tencent blockchain service [EB/OL]. https://cloud.tencent.com/product/tbaas/details.

[6] XUE Teng-Fei, FU Qun-Chao, WANG Cong, WANG Xin-Yan. A Medical Data Sharing Model via Blockchain. ACTA AUTOMATICA SINICA, 2017, 43(9): 1555-1562.

[7] WU Zhenquan, LIANG Yuhui, KANG Jiawen, YU Rong, HE Zhaoshui. Secure data storage and sharing system based on consortium blockchain in smart grid [J]. Journal of Computer Applications, 2017, 37(10): 2742-2747.

[8] Gu Jun, Xu Xin. Design and Implementation of a Humanities and Social Sciences Data Sharing Model: A Case Study of Consortium Blockchain. Journal of the China Society for Scientific and Technical Information, 2019, 38(4): 354-367.

[9] Xia Q, Sifah E, Smahi A, Amofa S. BBDS: Blockchain-Based Data Sharing for Electronic Medical Records in Cloud Environments [J]. Information, 2017, 8(2): 44.

[10] Tian F. A supply chain traceability system for food safety based on HACCP, blockchain & Internet of things [C]. 2017 International Conference on Service Systems and Service Management. 2017: 1-6.

[11] ZHOU Jie1, LI Wenjing. Research on logistics block chain consensus algorithm based on cloud computing [J]. Computer Engineering and Applications, 2018, 54(19): 237-242.

[12] BigchainDB GmbH. BigchainDB 2.0 The Blockchain Database [EB/OL]. (2018) [2019-7-15]. https://www.bigchaindb.com/whitepaper/bigchaindb-whitepaper.pdf.

[13] Svetlana S, Angelika K, Ifigenia G. The relationship between Bitcoin returns, volatility and volume: asymmetric GARCH modeling [J]. Journal of Enterprise Information Management, 2022.

[14] Savage N. The Outlook for Crypto [J]. Communications of the ACM, 2023.

[15] Schuh F, Larimer D. Bitshares 2.0: General overview [EB/OL]. (2017) [2019-7-15]. https://cryptorating.eu/whitepaperw/BitShares/bitshares-general.pdf.

[16] Hyla J, Su M A. Energy-Efficient Raptor-like LDPC Coding Scheme Design and Implementation for IoT Communication Systems [J]. 2023.