Research of Electronic Medical Record System Based on Blockchain

Lizhou Deng[1], Teng Long[1], Yuan Wang[1(✉)], Jianmao Xiao[1], and Li Fang[2]

[1] School of Software, Jiangxi Normal University, Nanchang 330022, Jiangxi, China
jxnuss@jxnu.edu.cn
[2] Ji'an Central Hospital, Ji'an 343000, Jiangxi, China

Abstract. The traditional medical record system is stored in the hospital side, which is not convenient for data sharing and transmission between hospitals; Patients can see the medical records data, therefore it is challenging to maintain patient privacy. Based on this, a disaster-tolerant Alliance blockchain is used to construct a distributed electronic medical record system that includes a peer-to-peer (P2P) network with mutually lengthy links and improved Practical Byzantine Fault Tolerance (I-PBFT). The system encourages medical data exchange and sharing, makes the flow of medical data transparent to patients, and transfers ownership of medical records back to patients, turning medical data into a valuable data asset for patients.

Keywords: Alliance Blockchain · P2P · Consensus Plugin · Medical Records Systems

1 Introduction

The amount of medical data has significantly increased with the growth of the medical business, particularly under the effect of COVID-19, making administration and preservation of medical data particularly crucial. Due to the importance of medical records as a source of data, several hospitals have developed electronic health records (EHR) for administration. The large volume and variety of medical record data are of great value to the diagnosis of patients' conditions and to the analysis and research of hospitals [1].

On the one hand, the disparate standards and file formats used by each hospital's electronic medical record system, along with the centralized storage approach, make it challenging to efficiently transmit medical record data between hospitals in today's Internet of Everything. In contrast, patients as users are fully unaware of how their personal medical information is used, which makes it challenging to protect its privacy and increases the danger of data shipping. Therefore, it is required to research a medical record system with secure storage, simple distributed storage sharing, and privacy protection in order to overcome these issues.

The blockchain technology can meet all these requirements at the same time. It was proposed in 2008 with features such as distributed storage and tamper-proof protection

Y. Cao and X. Shao (Eds.): MONAMI 2022, LNICST 474, pp. 3–17, 2023.
https://doi.org/10.1007/978-3-031-32443-7_1

[2], and has received a lot of attention and application worldwide for more than 10 years [3, 4]. Moreover, The state has given blockchain a lot of attention, and it is now part of the national information development plan [5], which has caused extensive research. Blockchain can be broken down into three main types based on application scenarios: public blockchain, private blockchain, and alliance chain.

The public blockchain is the most decentralized, and its typical applications are represented by bitcoin and Ethereum. Free from the control of any third party, each participant on the Internet (i.e., the nodes involved in consensus) is free to join as well as exit the blockchain network and to read the data records on the chain and participate in transactions as they wish. The fact that its operation is not controlled by a third party affects its use in the entity's organization. A small number of entities or people, whose access to data is constrained by organizational rules, govern and initiate consensus on private blockchains. Private chains are better suited for internal institutional use because of their rigorous access restrictions, despite the fact that transactions and consensus take less time. A stakeholder alliance can be formed in an alliance chain, which is a structure between public and private chains. The vetted nodes in the alliance have the power to fight for consensus ideas. Each participant in the chain is authorized to join the network. Obviously, the alliance chain is more suitable for transactions between different entities [6, 7]. Blockchain-based electronic medical record systems provide distributed storage to facilitate the sharing of shared medical record data, and their consensus mechanisms protect data privacy [8]. The electronic medical record system is designed with an Alliance chain structure and requires regulated data access, node addition and deletion, and other actions between patients and other hospitals.

A storage structure combining Merkle trees and directed acyclic graphs (DAG) is used to ensure the security of medical record storage, and a consensus mechanism using blockchain is used to protect the privacy of patient medical data. We have built a distributed medical record system using the Alliance chain to facilitate sharing among hospitals.

Building a disaster-tolerant de-primary node Alliance chain, building a peer-to-peer (P2P) network based on the T-io framework, designing distributed storage based on Merkle and DAG, and improving the Practical Byzantine Fault Tolerance (PBFT) consensus plugin are some of the main contributions made in this paper.

Organization. The remainder of the paper is organized as follows. Section 1 is an overview of this paper, and the related work is described in Sect. 2. Section 3 describes the paper's overall framework and main contribution. Section 4 briefly introduced a distributed storage design based on Merkle and DAG. We introduced P2P network based on the T-io framework in Sect. 5. Section 6 we propose I-PBFT consensus algorithm based on PBFT. The simulation experiment setup and results are presented in Sect. 7. Finally, in Sect. 8, we summarize the conclusion and future work.

2 Related Work

2.1 Blockchain Electronic Medical Records

The main goal of medical data platforms is to make it easier for doctors and patients to use and share data, including querying data on patients' symptoms, tests, and medications.

However, each platform currently has its own set of standards for collecting data, which are mostly fragmented, heterogeneous, and semi-structured, and these imperfect data add to the burden of integrators [7].

As blockchain technology advances, its study and use in the healthcare industry are gradually growing. Yang et al. [8] designed a blockchain-based EHR system to prevent data tampering and misuse by tracking all events that occur in the database. Dagher et al. [9] proposed a medical record system framework that gives ownership and control of the EHR to the data owner and uses multiple smart contracts to protect data privacy, and provide a secure and reliable medical record access process for patients and institutions. Wang et al. [10] proposed a blockchain-based EHR sharing approach with searchable symmetric encryption to improve privacy leakage and control issues during medical record sharing. Zhang et al. [1] implemented a dual-chain form of storage for medical record data, using a combination of attribute encryption and multi-keyword encryption to achieve accurate retrieval of data, to anonymously implement data sharing between different groups and achieve traceability. In May 2020, XENIRO on a mobile web edge server [11] will be released. It is based on the development of SnapScale, a platform for medical image data interaction with patient-centric access control and trusted storage for data sharing, the use of AI algorithms to perform auxiliary diagnosis on image data, and the encryption of the diagnosis results on the chain.

2.2 Consensus Plugin

The well researched consensus technique is employed in distributed environments to ensure that nodes concur to confirm the system state. Bitcoin system adopts the whole network consensus method based on proofs of work (referred to as POW [12]), which can resist the double spend attack under a certain probability.

However, POW algorithm has some problems such as low throughput and high resource consumption. A series of advanced protocols have emerged, such as proof of stack (referred to as POS [13]), which improves throughput and reduces resource consumption. So, resource concentration will affect fairness.

In addition, pease et al. Consider the fault tolerance of the system. The famous Byzantine fault tolerance (BFT) algorithm is proposed, but it has the problems of excessive network overhead and high resource occupation. Castro et al. [14, 15] first proposed PBFT algorithm can meet the tolerance of no more than 1/3 of Byzantine nodes and is adopted by fabric 0.6 project under Linux foundation. The PBFT method does, however, have a performance issue. A number of enhanced algorithms have been put forth by researchers to decrease communication costs and increase election security.

In the alliance blockchain, it is necessary to consider the existence of Byzantine nodes and the number of nodes is controllable, PBFT consensus algorithm is more appropriate. In PBFT algorithm, the block proposal node is always the main node. If the main node is not updated for a long time, other nodes cannot get the opportunity to be elected as the main node to obtain system incentive, which has a certain centralization risk and is not conducive to the long-term operation of the alliance system. Moreover, with the increase of the number of nodes, the PBFT communication overhead increases Huge [16].

To sum up, the PBFT method has to be improved to increase the reliability of master node selection, optimize the consensus procedure, decrease communication cost, and

decrease the algorithm's temporal complexity. In this study, a distributed electronic medical record system has been constructed based on a disaster-tolerant alliance chain.

3 Overall Framework

The data layer, persistence layer, network layer, and consensus layer are the four primary divisions of the distributed electronic medical record system based on blockchain. The data center, business service layer, data access layer, and business model layer are the divisions that can be made from the standpoint of system development, and Fig. 1 depicts the entire system architectural diagram.

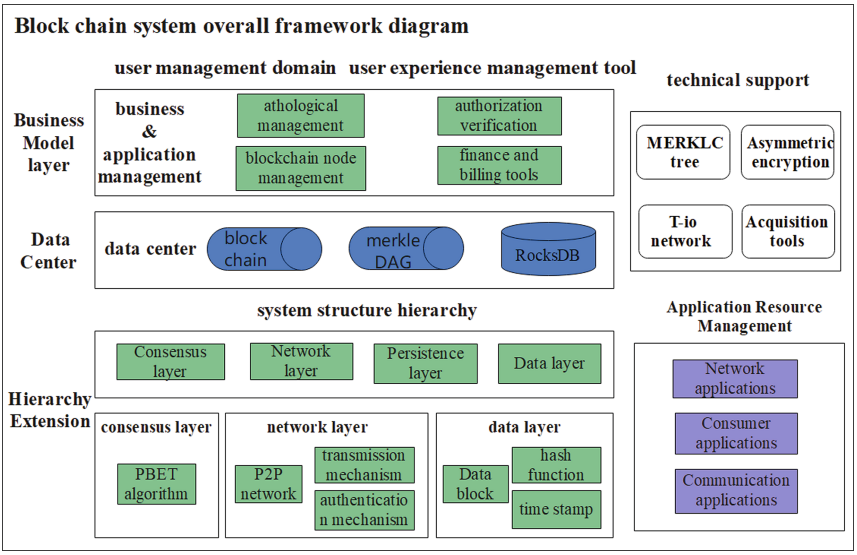

Fig. 1. Overall system architecture of the electronic medical record system

As shown in Fig. 1, the blockchain perspective includes the data layer, the persistence layer, the network layer, and the consensus layer.

- **Data Layer.** Consists of Data Blocks, Chained Structures, Hash Functions, Merkle Trees, Asymmetric Encryption, and Timestamps.
- **Persistence Layer.** The data for the project is primarily stored in this layer. The hospital node server information and log information of this project are primarily stored in the relational database MySQL since MySQL is lightweight and open source, making it easier to handle the node server information by utilizing MySQL. Among the additional essential data held on RocksDB are blockchain and medical record data, with the latter having been segmented and hashed for security.
- **Network Layer.** T-io network framework serves as the foundation for the network layer. T-io uses a network infrastructure called Asynchronous Blocking IO (AIO),

which has good translation performance. T-io is thought to utilize system resources more effectively in a network with many long connections and also includes grouping features, making it ideal for federated chains.
- **Consensus Layer.** The PBFT consensus plugin, a more effective consensus algorithm that is well suited for the establishment of the Alliance chain, was selected as the consensus layer. And in this project, we have improved this algorithm.

In addition, from a system architecture perspective, the specific layers of the system correspond to the following.

(1) **Business Model Layer:** Corresponds to the entity of each table in the database.
(2) **Data Access Layer:** It is used to access the database, operate the database and realize the persistence of data.
(3) **Business Service Layer:** The business service layer references the corresponding DAO database operations, encapsulates the requests of the representation layer, and each request is encapsulated as a method of the Service class; and the layer provides the corresponding interface for the Rest (Controller) layer to call.
(4) **Data Center:** Refers to the database and file system.

4 Distributed Storage Design Based on Merkle and Dag

This paper uses distributed storage technology in conjunction with blockchain to store data because of the importance and privacy of medical data. More specifically, Merkle and DAG are paired to swiftly delete duplicate data by the same hash value in order to reduce storage space, utilize hash ID to uniquely identify the content of a data block against tampering, and slice a whole data into numerous blocks. Assembling data can be done using the Kademlia (KAD) method, while downloading data can be done concurrently with the aid of hash addressing and downloading.

4.1 Storage Structure of Merkle and DAG

- **Storage method.** The file is divided into several pieces, and the hashing of each piece produces a distinct ID that can be quickly recognized and de-duplicated in the storage network. In order to maximize transmission efficiency, a block is typically kept in several copies on various network nodes.
- **Content Addressing Method.** Because each block has a distinct ID, you only need the node's ID to find the corresponding block. The Data Block holds the hash values of the files, while the Merkle Tree, also known as a Hash Tree, is a hash-based data structure in which the hash values are mostly kept. A concatenated Hash string of a node's associated children makes up a non-child node.

4.2 KAD Addressing Algorithm

The Kademlia (KAD) algorithm, which provides a decentralized file query system, is primarily a distributed hash table (DHT) based on an XOR distance algorithm. KAD uses both node hash and object hash addressing and uses XOR to determine the distance between hashes. The XOR (specificity or operation by ratio) technique is used to determine how far apart the keys are from one another.

The P2P node is the leaf node at the end of the 160-level binary tree created by KAD, and each node's location in the tree is determined by its 160-bit node hash. Whether a node is in the left or right sub tree of the tree is determined by the two potential values of each bit (0 or 1).

The KAD algorithm in our medical record system distributes data blocks with various hashes to nearby network nodes determined by XOR for the purpose of distributed storage. We just need to download the block data from the associated network node when requesting the file using the KAD technique, which is based on Merkle and DAG as well as the Hash value of each block. Finally, just verify that the data is complete and finish stitching. The blockchain data structure, Block, is designed as Table 1.

Table 1. Blockchain data structure

type or value	name
blockHeader	BlockHeader
blockBody	BlockBody
hash	String
calculateHash()	blockHeader.toString() + blockBody.toString()

4.3 Storage Design Implementation

The creation of the blockchain's most fundamental data structure serves as the foundation for implementing the data layer storage function. After unifying the data structure, by building the SpringBoot development framework, we construct the medical data storage function based on three technologies: the blockchain, relational databases, Merkle and DAG. Block, the blockchain's data structure, is created as follows.

5 The P2P Network Based on T-io Framework

We will use P2P network architecture to construct the blockchain network layer based on the data privacy and node correlation of the blockchain electronic medical record system presented in this study. We suggest that while choosing nodes for a P2P network, hospitals and medical research facilities should be considered. According to the alliance chain conceptual design, the P2P network system will chose or choose a trusted node as a manager to create an alliance chain, and this trusted node will be responsible for doing a lot of management tasks. To maintain the integrity of the entire P2P network, new nodes may be added and existing nodes may be removed. The trusted node connects to each configuration node it finds in the database when the P2P network is launched. A node officially enters the P2P network and begins connecting other nodes once it has been successfully joined by a trusted node. The trustworthy node will determine if a node is invalid and restrict it from connecting to the P2P network if it is unable to connect. After that, until the failed node is linked to the trusted node, the trusted node frequently makes connection requests to the failed node.

5.1 P2P Network Characteristics of Applicable Alliance Chain Based on T-io Architecture

The P2P network uses the T-io framework, which is based on the AIO design, to accommodate the prolonged connection and separation times between each node in this medical record system. The performance is excellent, resource consumption is modest, and group function is maintained when maintaining a high number of heartbeat packets across long connections. SaaS platforms with several alliance chains are therefore especially well suited for it. In this project, each node serves as both a client and a server. It will link to additional $N - 1$ nodes if it serves as a server. If the server is a client, it is connected to other $N - 1$ nodes. Every time you wish to send a message, create a Group in the same federation chain and call the send Group method directly.

The Fig. 2 shows the flow of the application layer enabling the kernel process to the kernel and returning a response. When there are many lengthy connections, the performance is good, the server's total resource usage is very low, and the grouping function is there. This makes it ideal for P2P network alliance chains. When the assigned resources are used up, any process in the network can use the AIO technique to directly request resources from other processes without the use of an event pool or resource monitoring. As a result, sustaining heartbeat packets with a lot of lengthy connections has good performance and a tiny resource footprint.

5.2 P2P Network Nodes Distributed Storage——KAD Algorithm

Based on KAD algorithm, we set a 160bit ID identifier for network nodes. In the algorithm, each node joining the network will be assigned a 160bit node ID (node ID), which is randomly generated. The key/value pair data is simultaneously stored on the nodes whose ID value is closest to the key value.

In light of this, XOR processing can be used to calculate the logical distance between Kademlia-based network nodes. An ID's key is associated with a data structure that

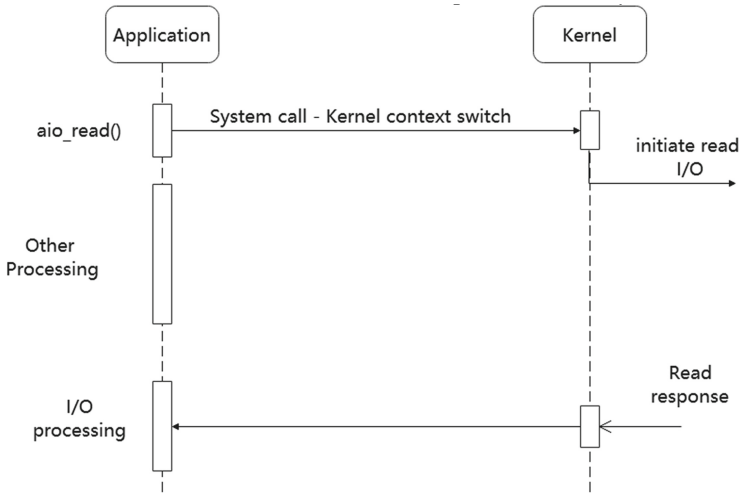

Fig. 2. AIO asynchronous non-blocking I/O mode adopted by T-io

resembles a binary tree. The n level of the binary tree corresponds to the NTH bit. A 1 or a 0 is processed for this bit, with the left subtree receiving the result. In the end, an ID binary tree is created.

Every item has a distinct binary ID, and the ID XOR operation reflects the bit differences in the ID. The ultimate difference outcome is more heavily influenced by the bits the higher they are. A binary tree is created if all of these IDs are stacked, with the highest bits on top and the lowest bits at the bottom, with the lower bits all on the same branch. The difference between the leaves (the ends of the branches) of the tree is represented by the difference between the IDs.

When dealing with 160bit network node ID, the storage space can reach 2l 60. Therefore, binary tree splitting is carried out. For route queries, the tree is broken up into numerous smaller trees that are stored on each node. For each 0 5 i 5 160, each node stores some node information within the range [2i, 2i + 1), which is composed of some data lists. The information in each of these lists is kept in chronological order according to the most recent time it was seen, with the most recent information at the head and the latest information at the tail of each K-bucket. There are no more than K data items in each bucket.

When you initially start, the entire ID digital region is often contained in just one K-bucket. When detected a node, we calculate the distance from ourselves and add it to the existing corresponding K-bucket:

Algorithm 1: K bucket.

Input: a new node $node_i$;
K - bucket;

Output: K -*bucket;*

1. **if** K - *bucket is not full* **then**
2. Add $node_i$ to the end of K — *bucket;*
3. **else if** *K-bucket is full and the K — bucket contains* $node_i$ then
4. the *K-bucket* splits downward *node branch direction* until it is no longer separable (i.e., i=0)*;*
5. **else**
6. take out the earliest inserted node and start Ping Node again*;*
7. **if** *there is no response* then
8. delete the node and add $node_i$ to the end of *K -bucket;*
9. **end**
10. **end**

In order to guarantee that searches eventually converge, KAD tries to store as many "nodes closer to it" routing designs as it can. The P2P network is successfully completed without the involvement of the central node to finish the address search as a result of our innovative usage of binary tree ID storage in the absence of a central server in the address operation after splitting into k-buckets.

6 De -Masternode Federation Chain Technology Based on I-PBFT Algorithm

6.1 I-PBFT Algorithm Without Elections

Because the federation chain itself exists in a trusted environment, any node can generate blocks and broadcast over the entire network, so the standard PBFT technique is enhanced without picking the leader in this study. Other nodes enter the preparation state after receiving a Block request and check the format, hash, and signature. They move into the prepare state and full network broadcast state after the verification is successful. When the accumulated number of Prepare for each node is greater than $2f + 1$ (f is the tolerable number of Byzantine nodes), the system enters the COMMIT state and broadcasts the status throughout the network. Each node believes it has reached consensus and adds the Block to the Block chain before executing the SQL statement in the Block when the Commit number it has accumulated is more than $2f + 1$ (f is the number of tolerable Byzantine nodes). The Leader is eliminated, resulting in a speedier and less resource-demanding algorithm overall. For the I-PBFT algorithm, the process is as follows:

1 A node 1 in the network broadcasts the transaction requested by the client to the whole network. All nodes except node 1 hash and broadcast across the network.
2 Each node performs a hash calculation and broadcasts to the entire network.

3 Assuming that a node receives $2f$ (f is the tolerable number of Byzantine nodes) hash equal to itself sent by other nodes, commit to the whole network.
4 If a node receives $2f + 1$ (f is the tolerable number of Byzantine nodes) commit messages (including itself), it can submit a new block to the local blockchain and state database.

Every node is equivalent and performs better. Three phases make up the I-PBFT consensus algorithm without elections: the voting queue, the submission queue, and the consensus algorithm. The following diagram illustrates how each stage is implemented in pseudo-code. Naturally, the method will save the hash set of the node's confirmed status during the voting phase, i.e., whether the node broadcast a commit permit or deny message.

After each node in the commit queue algorithm receives more than $2f + 1$ commit messages from different nodes (including itself), the block is considered to have reached a committed state and persisted to the blockchain database. The algorithms of voting, submitting and consensus are described as algorithm 2, 3 and 4 respectively.

Algorithm 2: voting algorithm

Input: *voteMsg* :voting queue;
hash : hash value of confirmed state (possibly null);
VoteMsgtemp : 1;
key : key value in *voteMsg* ;
number : the block number of the vote that has been received;

```
1.  voteMsg.get(hash) ;
2.  if voteMsg is null then
3.    voteMsg.put(hash,voteMsg) ;
4.  else
5.    for VoteMsgtemp : voteMsg do
6.      if VoteMsgtemp.IP == voteMsg.IP then
7.        return;
8.      end
9.    end
10. for key :VoteMsg, keySet(1) do
11.   if hash == key then
12.     continue;
13.   end
14.   if voteMsg.get(key).getNumber() < number then
15.     continue;
16.   end
17. end
```

Algorithm 3: submitting algorithm

preMsgQueue : submitting queue;
voteMsg : voting queue;
hash : hash value of confirmed state (possibly null);
VoteMsgtemp : 1;
key : key value in *voteMsg* ;
number : the block number of the vote that has been received;

1. *preMsgQueue* arrive;
2. *voteMsg* arrive;
3. *hash* = *voteMsg* .getHash();
4. **if** *voteMsg is null* **then**
5. *voteMsg.put*(*hash*, *voteMsg*) ;
6. **else**
7. **for** *VoteMsgtemp* : *voteMsg* **do**
8. **if** *VoteMsgtemp.IP* == *voteMsg.IP* **then**
9. return;
10. **end**
11. **end**
12. **for** key : VoteMsg, keySet **do**
13. **if** *hash* == *key* then
14. continue;
15. **end**
16. **if** *voteMsg.get*(*key*).*getNumber*() < *number* **then**
17. continue;
18. **end**
19. **end**

Algorithm 4: consensus algorithm

Input: *key* : key value in *voteMsg* ;

Output: *blockHashes* ;

```
1.  List <BlockHash> blockHashes = get( key );
2.  Map <String, Integer> map=new HashMap<>();
3.  for BlockHash blockHash: blockHashes do
4.      String hash= blockHashes.getHash() ;
5.      map.merge(hash,1,(a,b)->a+b);
6.  end
7.  String hash=getMaxKey(map);
8.  Return blockHashes ;
```

7 Experimental Performance and Simulation Analysis

This article evaluated the system, rapid link, and LAN Fabric in the local LAN environment. The following diagram illustrates the precise evaluation environment and configuration: Three nodes, a CPU with four 2.0 GHz cores, eight gigabytes of RAM, and CentOS 7.2 make up the configuration.

The transaction throughput of the Blockchain network serves as the performance efficiency evaluation index for Blockchain. The maximum number of transactions that may be performed in a given amount of time is known as transaction throughput:

1) Suppose N_t is the number of transactions sent in the statistical time and N_b is the number of transactions recorded on the block in the statistical time, and the transactions are randomly generated by a transaction generator with adjustable frequency to satisfy N_t controllability.
2) When measuring the transaction throughput, random transactions are concurrently generated at a certain rate. The transaction generation rate is continuously increased until N_t is greater than N_b, and the configuration parameters of the piezometric system at this time are recorded, which are the optimal piezometric parameters.
3) According to the optimal pressure testing parameters, the blockchain system is stress tested for several times, and the number of transactions recorded on the block per unit time is recorded and averaged, which is the transaction throughput.Transaction throughput is mainly tested on one metric: invoking contract TPS, TPS can be expressed as Eq. (1).

$$TPS = transactions/\Delta t \tag{1}$$

where transactions is the number included in the blockchain over a period of time. And Δ_t is the recording time, which is generally an integer multiple of the block generation time. The TPS comparison is shown as Table 2 (Fig. 3).

Table 2. TPS Comparison

	chain	Fabric	I-PBFT
TPS (pen/sec)	*388*	1368.7	1599.6

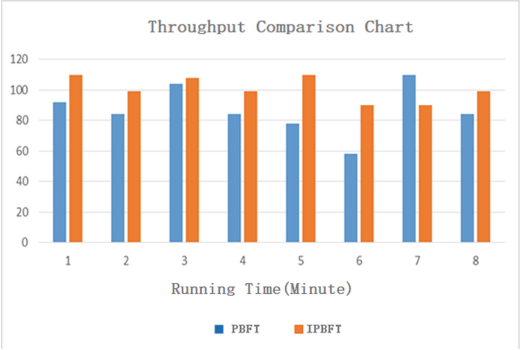

Fig. 3. Throughput Comparison Chart

Using the transaction confirmation rate as the evaluation metric, we compare the operational efficacy of the PBFT consensus mechanism and the I-PBFT consensus mechanism proposed in this paper. We also test the performance differences between the two algorithms under various ratios of false nodes to total nodes and under various running times. The outline is depicted in Fig. 4.

Figure 5 shows the change of transaction confirmation rate of PBFT and I-PBFT over a long period of time.

With PBFT and its derivate consensus processes, there is a communication barrier between nodes that must be overcome. In the new PBFT, the check-pointing protocol does not call for additional communication because the Leader election mechanism has been eliminated, and the system starts up quickly or has fewer error nodes. On the other hand, this decreases the volume of data while also reducing the view switching protocol

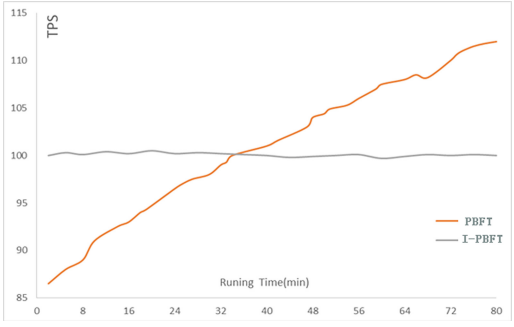

Fig. 4. Comparison of changes in Long-term throughput rate Chart

calls in scenarios where there are more error nodes. The graph compares the volume of data sent between nodes. The system duration is plotted on the horizontal axis, while the number of P2P communications with complexity O(n2) needed to construct a block is plotted on the vertical axis.

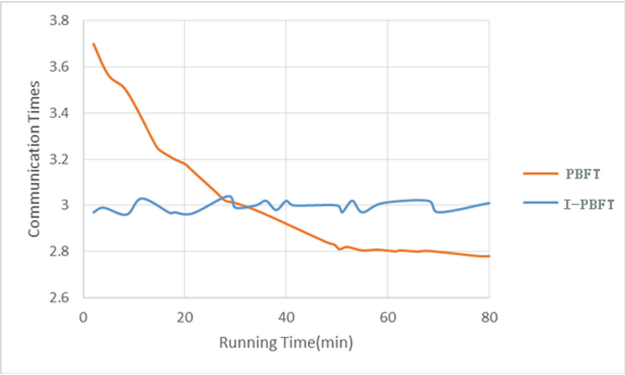

Fig. 5. Comparison of changes in Construct block per unit time rate Chart

8 Conclusion

Electronic medical record systems have demonstrated a tendency toward patient-centeredness in recent years with the promotion and deployment of cloud computing and big data technologies, where patients have more authority over their own medical records. In accordance with this paradigm, people maintain their medical records on cloud servers and selectively share medical information through controlled access with physicians or other third-party providers.

The development of blockchain technology has made decentralized data storage possible. Referring to various well-known blockchain systems including Bitcoin, IPFS, and Ethereum, this system has taken a year. At the same time, we customized the blockchain system for the storing of medical data, discovered more blockchain applications through exploration and study, and made numerous enhancements to the system as it stood at the time. The distributed data storage is the project's major challenge, and few papers and Internet sites provide comprehensive information about it.

In the future, we will concentrate on the development of medical data and transaction security mechanisms to offer a secure, user-friendly, and convenient environment for the value co-creation of medical data, to better promote patient control of their medical data and to promote people's health, and to fairly mine the potential value of medical data.

Acknowledgment. This research is funded by the Natural Science Foundation of Jiangxi Province under Grant No. 20192ACBL21031, by the Science and Technology Research Project of Jiangxi Provincial Department of Education (No. GJJ181492), by the Higher Education Research Project on Educational Reform in Jiangxi Province (No. JXJG19221).

Data Availability. The data, including block and performance indicators in the experiments, used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest. The authors declare that they have no conflicts of interest.

References

1. Zhang, L., Zheng, Z.Y., Yuan, Y.: A controllable sharing model for electronic health records based on blockchain. Acta Automatica Sinica **47**(9), 2143–2153 (2021)
2. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. http://www.bitcoin.org/bitcoin.pdf (2008)
3. Kang, J., Xiong, Z., Niyato, D., Ye, D., Kim, D.I., Zhao, J.: Toward secure blockchain-enabled internet of vehicles: optimizing consensus management using reputation and contract theory. IEEE Trans. Veh. Technol. **68**(3), 2906–2920 (2019)
4. Dinh, T.T.A., Liu, R., Zhang, M., Chen, G., Ooi, B.C., Wang, J.: Untangling blockchain: a data processing view of blockchain systems. IEEE Trans. Knowl. Data Eng. **30**(7), 1366–1385 (2018)
5. Wu, M., Wang, K., Cai, X., Guo, S., Guo, M., Rong, C.: A comprehensive survey of blockchain: from theory to IoT applications and beyond. IEEE Internet Things **6**(5), 8114–8154 (2019)
6. Mao, H.-Y., Nie, T.-Z., Shen, D.-R., Yu, G., Xu, S.-C., He, G.-Y.: Survey on key techniques and development of blockchain as a service platform. Comput. Sci. **48**(11), 4–11 (2021)
7. Jacob, F., Mittag, J., Hartenstein, H.: A security analysis of theemerging p2p-based personal cloud platform Maidsafe. In: 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, pp. 1403–1410 (2015)
8. Liu, J., Li, X., Ye, L., et al.: BPDS: a blockchain based privacy- preserving data sharing for electronic medical records. In: GLOBE- COM 2018 IEEE Global Communications Conference. IEEE (2018)
9. Dagher, G.G., Mohler, J., Milojkovic, M., et al.: Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. Sustain. Cities Soc. **39**, 283–297 (2018)
10. Wang, S., Zhang, D., Zhang, Y.: Blockchain-based personal health records sharing scheme with data integrity verifiable. IEEE Access **07**, 102887–102901 (2019)
11. Chuang, I.H., Chiang, S.H., Chao, W.C., et al.: A hierarchical blockchain-based data service platform in MEC environments. In: ACM 2nd International Conference on Blockchain Technology, 12–13 Mar 2020, pp. 95–99. Association for Computing Machinery, Hilo, USA (2020)
12. Jakobsson, M., Juels, A.: Proofs of work and bread puddingprotocols. In: Ifip Tc6/tc11 Joint Working Conference on Secure Information Networks: Communications & Multimedia Security Kluwer, B.V. (1999)
13. King, S., Nadal, S.: PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake (2012)
14. Castro, M., Liskov, B.: Practical byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst. **20**(4), 398–461 (2002)
15. Xu, G., Liu, Y., Khan, P.W.: Improvement of the DPoS consensus mechanism in blockchain based on vague sets. IEEE Trans. Industr. Inf. **16**(06), 4252–4259 (2019)
16. Qian, W.N., Shao, Q.F., Zhu, Y.C., Jin, C.Q., Zhou, A.Y.: Research problems and methods in blockchain and trusted data management. J. Softw. **29**(1), 150–159 (2018). (in Chinese)