An Improved blockchain-based secure medical record sharing method

Hüseyin Bodur (✉ huseyinbodur@duzce.edu.tr)
Duzce University

Research Article

Keywords: Elektronic medical records (EMRs), healthcare, blockchain, security

Posted Date: November 17th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2109937/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License

Additional Declarations: No competing interests reported.
An Improved blockchain-based secure medical record sharing method

Hüseyin Bodur

Department of Computer Engineering, Düzce University,
Konuralp, Düzce, 81000, Turkey.

Contributing authors: huseyinbodur@duzce.edu.tr;

Abstract

Today, the confidentiality and security of patient medical records is of great importance. A database where patient records are kept should not fall into the hands of malicious persons. In addition to keeping medical records confidential and secure, their sharing among various entities such as individuals or institutions must be auditable and controllable. This study proposes a method that enables the storage, access, and sharing of medical data without risking security vulnerability. In the proposed method, data are stored on the cloud, whereas data access information and hash values are stored on a blockchain. The three consensus mechanisms (Proof of Work (PoW), Proof of Stake (PoS), and Proof of Authority (PoA).) for the proposed scheme are implemented and compared with studies in the literature in terms of security analysis. The performance analysis of the proposed method showed that the Proof of Work gives the best result in terms of block size, while the Proof of Stake gives the best results in terms of block creation time and memory usage. The security analysis showed that the proposed method has many security features and is strong against attacks.

Keywords: Electronic medical records (EMRs), healthcare, blockchain, security

1 Introduction

Data processing is of great importance in the healthcare industry. It is necessary to apply the right treatment by correctly evaluating sensitive patient
data. Today, patient records are kept electronically. These Electronic Medical Records (EMRs) are basically data or records of the patients in a hospital management system. The EMRs are collected by healthcare organizations such as hospitals and clinics where they are used to store general information such as patient treatment plans, medical histories, and ultrasound and other images. Although patient EMRs are kept in a digital environment, this environment is generally local. In addition, the authority over EMRs usually belongs to the hospitals. Although patients have the right to access their own EMRs, the authority to share their data is unclear.

Hospital EMRs consist of large volumes that need to be stored securely, processed effectively, and shared without delay. Keeping EMRs in a public database to create big data ensures comprehensive access to patient data. In this way, a patient’s medical records can be evaluated by different entities. Evaluation of EMRs by processing them with different entities can increase the variety and accuracy of treatment methods. In addition, a patient’s EMR can be evaluated by researchers who are far from the patient. Therefore, cloud servers are used to process EMRs effectively.

Cloud servers are a good option for storing and processing big data quickly. However, storing EMRs directly on a cloud server has disadvantages in terms of security and privacy. Because EMRs contain sensitive data, they should not fall into malicious hands and should not be used without patient permission. A cloud managed by a third-party service provider is semi-trustworthy, i.e., stealing or tampering with user data may occur [1] and data can be leaked through malicious attacks on a cloud server [2, 3].

Blockchain technology possesses features such as data integrity, consistency, and a protection mechanism against alterations [4]. Thus, it is difficult for malicious individuals to damage the data kept in blocks on a blockchain. Leveraging a blockchain to ensure the confidentiality and security of sensitive patient EMRs provides an advantage in terms of data integrity.

In this study, a method was developed for the secure storage and rapid sharing of sensitive patient EMRs. A distributed database and a blockchain are used together in this approach. Although patient EMRs are kept in an encrypted distributed database, a blockchain is used to ensure data integrity. In the proposed method, all authority over a patient EMR rests with the patient. An authorization process for accessing patient data is carried out via an application. In addition to patients, the application includes entities such as hospitals and third-parties. Keeping patient EMRs on a cloud server enables the data to be easily accessed and processed quickly. This also allows researchers to analyze a patient’s EMR from anywhere, regardless of time and place.

2 Related Works

There are various studies in the literature relating to the storage, processing, and sharing of patient data without compromising security and privacy. In
one of these studies, Zyskind et al. [5] proposed a peer-to-peer network called Enigma which allows different parties to collectively store and evaluate the data, while keeping the data on the network confidential. In their work, an external blockchain was used to control network access, to manage identity, and to prevent alteration of data content. In another study, Xia et al. [6] proposed a data-sharing framework that used the features of a blockchain to address the problem of accessing data stored on the cloud. In the proposed method, access to data is provided only by invited and known users, with the aim of preventing malicious access.

In another study, Yue et al. allowed patients to possess and control their own data using a blockchain platform in which patient data were encrypted and kept. Patients were made aware of third-parties that might have access to their data [7]. Xia et al. designed a data-sharing model between cloud service providers using a blockchain. In their MeDShare model, smart contracts and access control mechanisms are used to continuously monitor activities concerning the data and to revoke access permission if necessary [8]. In another study, Johari et al. developed a blockchain simulation in Python in which a list of patient records is kept in each block [9]. In their study, Chen et al. developed a medical information system model based on a blockchain that collects real-time patient data from medical devices during a surgical procedure. The model enables third-party users to access secure data via cloud servers through various applications [10].

Roehrs et al. proposed a distributed model called OmniPHR that would collect personal health records at one point and then determine how healthcare providers would access patient data. The model enables patients to manage their health data and health providers to share data with various health organizations [11]. In their study, Zhu et al. developed a blockchain-based Merkle tree for efficient and secure storage, transmission, and sharing of EMRs. The proposed scheme, unlike its original structure, uses a convolution operation to provide a significant reduction in the number of layers and the amount of data storage calculation [12]. In another study, Tang et al. developed a blockchain-based method by which patients and medical institutions could access medical records. In this approach, patient data are kept encrypted on the blockchain. The system enables a searching operation to be performed on encrypted patient data without the need for decryption. This ensures a high level of confidentiality and controllability [13].

In their study, Sharma et al. proposed a smart contract-based scheme to ensure the security and privacy of healthcare big data. They divided the proposed scheme into three main components: distributed application, a smart contract, and the InterPlanetary File System. They performed basic operations such as access control, data encryption, and storage via smart contracts [14]. Zhang et al. proposed a patient contact tracing scheme called PTBM that preserves privacy without violating the identity and location of a patient with infectious Covid-19. The PTBM is 5G-integrated and can effectively monitor patients and their corresponding close connections to identify those who
have been in possible contact with a diagnosed patient [15]. In another study, Nguyen et al. proposed a new electronic health record (EHR) sharing scheme by combining a blockchain with the InterPlanetary File System on a mobile platform. In their work, they presented an Ethereum blockchain that performs EHR transactions and a mobile application prototype connected to an Amazon server [16].

3 Blockchain Technology

The blockchain is a decentralized data management technology with a distributed consensus mechanism that eliminates the need for confirmation of peer-to-peer network transactions by a third-party [17]. This technology first emerged with Bitcoin, a digital currency proposed by Satoshi Nakamoto in 2008 [18]. The transactions on the network are verified and approved by the consensus of the majority of existing members in the network. This eliminates the role of a central confirming authority. The transactions are carried out using consensus protocols by obtaining the consensus of the majority of the members [19]. The confirmed transaction data are kept on a secure distributed ledger.

The ledger is a chained data structure. The data are encrypted and added to the chain in blocks. Unlike the traditional chain structure, each block contains the value of the previous block. In this way, a logical connection is formed between the blocks, and change or deletion of the blocks is prevented and thus strengthening the ledger against various attacks. As a global database, the ledger is accessible to everyone. Unlike traditional databases, it does not belong to any person or company. Every member in the network is a node. As seen in Figure 1, each member is connected to the network in a distributed way. The ledger is available at all nodes in the network and is updated periodically.

Blockchain technology possesses features such as data integrity, consistency, and a mechanism to protect against changes. In addition, interest in this
3.1 Blockchain Structure

Each block in a blockchain consists of two parts: the header and the body. As seen in Figure 2, the header of a block includes a timestamp, a Merkle root, a summary of the previous block, and a "nonce" value. The timestamp holds the current time information. The Merkle root has a binary tree structure and all transactions in the block body are contained in the leaves. The hash values of transactions are extracted. At each non-leaf node, the hash values of the child nodes are combined and the hash of that node is extracted, as expressed in the following equation:

\[ \text{Hash}_01 = \text{Hash}(\text{Hash}_0 + \text{Hash}_1) \]

This process continues until the root node. The hash value of the root node is the validation of all transactions. When any transaction on the tree is changed, its hash value also changes and then, all the hash values on the path from it to the root node will change. The validation process will fail when the root node hash has changed. The Merkle tree allows for secure and efficient validation of big data structures. Each new block added to the blockchain must have the hash value of the previous block. When creating a new block, the information in the header is hashed and the hash value is checked for an appropriate character set. If the hash value of the block has the appropriate character set, the block is accepted and added to the chain. In this way, a chain structure of blocks related to each other is created.

The new block information is also shared with all nodes in the network. If an appropriate block hash value is not found, the nonce value is changed and the process is repeated until the appropriate hash value is obtained. When the nonce value reaches a certain limit, if a block with a valid hash value still cannot be created, the other values that constitute the block header are updated and the flow is restarted. The hash value must have a predetermined difficulty level.
3.2 Consensus Mechanisms

In this section, three consensus mechanisms are briefly described, including Proof of Work (PoW), Proof of Stake (PoS), and Proof of Authority (PoA).

3.2.1 Proof of Work (PoW)

The PoW is the consensus mechanism originally used within the blockchain technology. All transactions made on the blockchain are recorded in an open ledger. Transactions are verified and executed with the consent of the majority of users in the network, rather than a third-party. This operation is called the PoW. The PoW is the most important feature of blockchain technology and because of its decentralization feature, interest in the technology has increased [19]. According to PoW, the majority of users on the network must approve the correctness of the transaction. For example, in the case of a new block to be added to the blockchain, if the majority do not agree, the transaction cannot be carried out. The PoW is an approach that requires the consent of a majority of a distributed user group. Miners are constantly producing blocks and are checking whether these blocks have been created properly [21]. They spend computer resources (time, power, and computing power) to establish a new block, which must be completed by solving complex calculation problems [22]. If a miner notifies the users on the network that it has created a correct block, the users must check whether this block is correct and should be added to the chain. If the majority of users in the network confirm that the created block is correct, it is added to the end of the blockchain and the miner is rewarded.

3.2.2 Proof of Stake (PoS)

The PoS is the consensus mechanism designed according to the concept of equity to replace the complex computational task in PoW. The user with the highest equity has the authority to verify the block. The equity is computed based on the user’s currency amount and the number of days of ownership. After each successful verification of a block, the equity is reset to zero and recomputed, and equity-based rewards will be given. The miners are called validators. There is no mining reward, and instead the validators are incentivized with transaction fees as there is no mining process. The PoS suggests buying cryptocurrencies instead of buying devices with high features such as extremely high resource consumptions to win in the block creation competition [23]. It aims to allow the users with large amounts of currency to jointly ensure the security of the blockchain and reduce the resource and time consumption of its building blocks. But it causes that the users with more currencies have higher probability and currency [24], [25].

3.2.3 Proof of Authority (PoA)

The PoA was proposed by co-founder of Ethereum in 2017. It is an energy-saving consensus algorithm based on the reputation of Byzantine Fault
Tolerance. Instead of staking currency as in PoS, users must prove their real identities. The purpose of PoA is to create an authorized block validator user for the person who created the block and is a centralized consensus mechanism compared to PoS. The PoA is collateralized by reputation to create the block. A limited number of users are trusted, pre-approved and allowed by the network, and are identified by a unique ID known as authorities [26]. All blocks will be created by selected authorized users, which reduces the time required to generate blocks and resolve data consistency issues. There is no incentive reward, rather if the user generates an invalid block, it behaves like a malicious user and is kicked off the authority list [27].

4 The Proposed Method

In this section, a method is proposed to ensure efficient and secure storage and sharing of patient EMRs. In the proposed method, a document-oriented database and a blockchain structure are used together. Whereas the patient EMRs are kept in the encrypted distributed database, the blockchain is used to ensure data integrity and security in accessing the data. Figure 3 shows the architecture of the proposed method.

4.1 Scenario of the Proposed Method

Five entities are involved in the system model: patients, hospitals, third-party applications, third-party entities, a document-based distributed database, and the blockchain.

4.1.1 Patients (p)

Patients are the owners of their EMRs. They have the authority to control their data. Patients can grant or revoke third-party entities access to their EMR through the third-party application. A smart contract is used to grant and revoke access.
4.1.2 Hospitals (h)

Patient EMRs, which are formed as a result of various examinations and diagnoses, are first recorded in the local databases of the hospitals. Hospitals create the record of a patient if the patient has no previous record in the system and then help the patient to encrypt and upload the EMR to the system.

4.1.3 Third-party entities (e)

Third-party entities are other hospitals, clinics, research institutions, doctors, health analysts, etc. They request permission from patients to access their EMRs through a third-party application. If the patient gives permission, they can access the data and carry out their research on it. In the proposed method, third-party entities are registered to the system through the application. A consensus mechanism is used to approve or revoke third-party entities, patients, and hospitals that register on the application. The consensus mechanism includes approval of entities other than third-party entities in the application seeking to join the network or needing to be revoked. The consensus mechanism is distributed. The approval transactions are performed by distributed users rather than by a central entity. The transactions occur with the approval of a certain percentage of users who take part in the consensus. The consensus mechanism is managed by a smart contract.

4.1.4 Document-Oriented Database

Patient EMRs are kept in the document-oriented database and are encrypted with a symmetric encryption algorithm. Although patients are the owners of their EMRs, they cannot make changes to their data, but can only authorize or remove access if requested. Each patient’s EMR is encrypted with that patient’s private key and kept in the database. In the database, apart from the patient EMRs, the data resulting from analyses of these EMRs by third-party entities are kept encrypted with the patient’s secret key. An EMR is kept on the database as follows.

```json
{
   "pid": "111342",
   "firstname": "Ahmet",
   "lastname": "Yilmaz",
   "address": "address data",
   "emr": {
      "data": "wetegbxfxygcerufjerufj674trsxgsgwr..."
   },
   "images": ["encrypted_1113421.png", "encrypted_1113422.png"]
}
```
4.1.5 Blockchain

When a patient EMR is recorded in a distributed database, a digest of the EMR is recorded in a block on the blockchain. Although the blockchain is open to everyone, it is not possible to change the blocks. In this way, the digest of an EMR cannot be changed in any way. By enabling any changes to be detected, this prevents alteration of the EMRs kept encrypted on the document-oriented database. A descriptive summary is also included in the blockchain that gives an overview of each patient’s disease and EMR. The descriptive summaries are unencrypted and contain key words about the patients. Using the descriptive summary, third-party entities can gain insight about a patient without accessing the patient’s sensitive EMR. The descriptive summaries are of great importance when requesting patients for permission to access their EMRs.

4.1.6 Third-party application

The application works as a front-end part and provides an interface for the users. Patients have the right to view their EMRs and authorize or deauthorize third-party entities through the application. In addition, third-party entities may request patients for access to their EMRs through the application. After examining and evaluating the EMRs they have accessed, they save the results they have obtained in the database through the application.

4.2 Specific Design of System Workflow

This section covers the specific design of the system workflow based on the proposed method.

4.2.1 Data Saving Phase

Patients register to the system through the application. After registration, each patient has a unique ID, a secret key created with a symmetric encryption algorithm, and two keys related with each other, one a public key and the other a private key, created with an asymmetric encryption algorithm: $p_id$, $p_id$secretKey, $p_id$publicKey and $p_id$privateKey.

The public key $p_id$publicKey is added to the policy list on the cloud. The policy list is a library where the public keys of the entities in the system are kept. This library is managed by a smart contract.

It is assumed that patients obtain their EMRs in a secure way by communicating with the hospital where their EMR is located. Patients encrypts and save their EMRs in the document-based database with their secret key: $encrypted_{EMR} = EMR_{(p_id)}secretKey$.

The hash value of the patient EMR is obtained with a hash function: $Hash(EMR_{p_id})$.

A descriptive summary is created that includes the patient’s disease and EMR: $diseaseDescription_{p_id}$.

The descriptive summary helps third-party entities gain insight about a patient without having access to the patient’s EMR. A new block is created
on the blockchain that contains the hash value and the descriptive summary. The block also contains the block id, $p_{id}$, timestamp, nonce value, and previous block hash.

### 4.2.2 Data Sharing Phase

Third-party entities are entities that are registered with the application and authenticated by the application. The authentication process is carried out through the consensus mechanism. These entities also have an ID and two keys related to each other, one a public key and the other a private key, created with an asymmetric encryption algorithm: $e_{id}$, $e_{id}\_public\_Key$ and $e_{id}\_private\_Key$.

A third-party entity can access disease descriptions contained in blocks via the blockchain. If the third-party entity wants to access a patient’s EMR after viewing the disease in the descriptive summary, it can communicate with the patient through the application. It sends the request message and the signed version of the request message with the secret key, together with its ID information:

$$request\_message = \{ e_{id}, message, messageHash(e_{id}\_private\_Key) \}$$

The patient receives the $e_{id}$ and first sends it to the policy list. The third-party entity is controlled through a smart contract. After controlling the $e_{id}$, the patient is sent the public key of the third-party entity:

$$PK_{e_{id}} \leftarrow (e_{id})\_public\_Key$$

The patient then verifies the third party’s signature using the public key of the third-party entity:

$$Hash(message) == messageHash_{PK_{e_{id}}}$$

If it is verified, the patient evaluates the request. If the request is approved by the patient and the third-party entity is authorized to access the patient’s EMR, the patient encrypts his/her secret key with the public key of the third-party entity and sends it to the third-party entity:

$$approved\_message = \{ p_{id}\_secret\_Key, PK_{e_{id}} \}$$

The third-party entity obtains the patient’s secret key $p_{id}\_secret\_Key$ from within the approved\_message using its private key. It decrypts patient EMRs with this key. After the third-party entity has completed the analysis of the data, the results, if any, are recorded in the database with the patient’s public secret key. Patients can access the analysis results using their own secret keys. After the third-party entity analysis of a patient EMR, the data are hashed and compared with the hash value on the blockchain. If the hash values are not equal, the data have been modified. In this case, the third-party entity along with its $e_{id}$ are revoked. The patient’s EMR in the database is also deleted. The patient then generates a new secret key, encrypts the EMR with this key and re-saves it in the database.

### 4.3 Evaluations

In this section, performance and security evaluations of the proposed scheme are given.
4.3.1 Performance Evaluation

In the study, blocks were created using PoW, PoS and PoA consensus mechanisms, respectively. These mechanisms were compared with each other in terms of block sizes, block creation times and memory usages. A few changes are required for the PoW, PoS, and PoA consensus mechanisms to be used in the proposed study. Firstly, the difficulty level chosen for the PoW mechanism is a simple pattern with the first three characters of the block header being “0”. It is aimed to keep the speed and performance of the proposed study high by keeping the difficulty level of block creation low.

Table 1  Block sizes in the proposed study (byte)

<table>
<thead>
<tr>
<th>B.C.</th>
<th>PoW</th>
<th>PoS</th>
<th>PoA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2°</td>
<td>251</td>
<td>277</td>
<td>462</td>
</tr>
<tr>
<td>2¹</td>
<td>476</td>
<td>530</td>
<td>799</td>
</tr>
<tr>
<td>2²</td>
<td>843</td>
<td>951</td>
<td>1468</td>
</tr>
<tr>
<td>2³</td>
<td>1657</td>
<td>1878</td>
<td>2838</td>
</tr>
<tr>
<td>2⁴</td>
<td>3295</td>
<td>3739</td>
<td>5588</td>
</tr>
<tr>
<td>2⁵</td>
<td>6577</td>
<td>7463</td>
<td>11090</td>
</tr>
<tr>
<td>2⁶</td>
<td>13129</td>
<td>14911</td>
<td>22098</td>
</tr>
<tr>
<td>2⁷</td>
<td>26271</td>
<td>29836</td>
<td>44172</td>
</tr>
<tr>
<td>2⁸</td>
<td>52600</td>
<td>59756</td>
<td>88205</td>
</tr>
<tr>
<td>2⁹</td>
<td>104810</td>
<td>119596</td>
<td>177036</td>
</tr>
<tr>
<td>2¹⁰</td>
<td>209728</td>
<td>239500</td>
<td>350452</td>
</tr>
<tr>
<td>2¹¹</td>
<td>420468</td>
<td>480569</td>
<td>708544</td>
</tr>
<tr>
<td>2¹²</td>
<td>841982</td>
<td>962620</td>
<td>1419933</td>
</tr>
<tr>
<td>2¹³</td>
<td>1685244</td>
<td>1926714</td>
<td>2842713</td>
</tr>
<tr>
<td>2¹⁴</td>
<td>3377842</td>
<td>3864904</td>
<td>5688273</td>
</tr>
</tbody>
</table>

The block creation process for PoS mechanism is based on the score values and the latest block creation dates of the validators in the authorized validator group. The score value represents the number of blocks that a validator has successfully created. The latest block creation date is calculated on a day-by-day basis by subtracting the current date. The stake value for PoS mechanism is calculated by multiplying the score value by the day value; score x days. The validator with the highest stake is selected to generate a new block. Then, the last block creation date is reset, preventing the validator from creating a new block for a while. The block creation for PoA mechanism is carried out sequentially by the validators within the authorized validator group whose identities have been proven before.

There are similarities in every block created with PoW, PoS and PoA mechanisms. Regardless of the consensus mechanism, each block contains the following data, which are similar to each other; “blockIndex”, “timestamp”, “previousHash”, “patientId”, “hashValueOfPatientEMR”, “diseaseDescriptionOfPatient”. Unlike other consensus mechanisms, a block created with the
An Improved blockchain-based secure medical record sharing method

PoW mechanism contains “nonce” data. A block created with the PoS mechanism contains the data ”validatorId”, ”validatorName” and ”validatorStake”. A block created with the PoA mechanism contains the data ”validatorId” and ”validatorPublicKey”.

Table 2 Block creation times in the proposed study (millisecond)

<table>
<thead>
<tr>
<th>B.C.</th>
<th>PoW</th>
<th>PoS</th>
<th>PoA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^0$</td>
<td>102</td>
<td>10</td>
<td>28</td>
</tr>
<tr>
<td>$2^1$</td>
<td>218</td>
<td>12</td>
<td>34</td>
</tr>
<tr>
<td>$2^2$</td>
<td>237</td>
<td>15</td>
<td>26</td>
</tr>
<tr>
<td>$2^3$</td>
<td>322</td>
<td>19</td>
<td>31</td>
</tr>
<tr>
<td>$2^4$</td>
<td>514</td>
<td>22</td>
<td>39</td>
</tr>
<tr>
<td>$2^5$</td>
<td>653</td>
<td>24</td>
<td>47</td>
</tr>
<tr>
<td>$2^6$</td>
<td>1221</td>
<td>30</td>
<td>71</td>
</tr>
<tr>
<td>$2^7$</td>
<td>1864</td>
<td>44</td>
<td>98</td>
</tr>
<tr>
<td>$2^8$</td>
<td>3147</td>
<td>53</td>
<td>123</td>
</tr>
<tr>
<td>$2^9$</td>
<td>6346</td>
<td>78</td>
<td>156</td>
</tr>
<tr>
<td>$2^{10}$</td>
<td>12868</td>
<td>104</td>
<td>215</td>
</tr>
<tr>
<td>$2^{11}$</td>
<td>24167</td>
<td>151</td>
<td>837</td>
</tr>
<tr>
<td>$2^{12}$</td>
<td>41362</td>
<td>253</td>
<td>1695</td>
</tr>
<tr>
<td>$2^{13}$</td>
<td>85129</td>
<td>507</td>
<td>3258</td>
</tr>
<tr>
<td>$2^{14}$</td>
<td>168185</td>
<td>848</td>
<td>5841</td>
</tr>
</tbody>
</table>

In the proposed study, $2^\text{n} (0 \leq n \leq 14)$ blocks were created by using PoW, PoS and PoA mechanisms, respectively. Consensus mechanisms were compared with each other in terms of block sizes, block creation times and memory usages. The results including the block sizes, block creation times and memory usages are shown in Tables 1–3, respectively. In the tables, B.C. means Block Count.

Table 1 presents block sizes of PoW, PoS and PoA mechanisms. According to these results, the PoA mechanism gave the highest result. The reason for this is that the public keys of the validators are stored in the blocks. The
public keys of the validators are used for proof in the authority process. The PoA mechanism was followed by PoS mechanism. The PoW mechanism gave the best result in terms of block size. Figure 4 gives a graphical representation of Table 1.

Table 2 presents block creation times of PoW, PoS and PoA mechanisms. According to these results, the PoW mechanism gave the highest result. The block must have a certain degree of difficulty for creating successfully in PoW mechanism. A successful block creation, called mining, requires high processing power. The PoW mechanism was followed by PoA mechanism. The PoS mechanism gave the best result in terms of block creation time. Figure 5 gives a graphical representation of Table 2.

Table 3 presents memory usages of PoW, PoS and PoA mechanisms. According to these results, the PoW mechanism gave the highest result. The high processing power required for successful block creation directly affected
memory usage. The PoW mechanism was followed by PoA mechanism. The PoS mechanism gave the best result in terms of memory usage. Figure 6 gives a graphical representation of Table 3.

4.3.2 Security Evaluation

The method proposed in the study of [14] includes an administrator. This leaves the system weak in the face of attacks against the administrator. If an attacker seizes administrator authorizations, the system becomes vulnerable to attacks. An EHR manager is at the center of the study of [16]. This EHRs manager evaluates requests and performs many important tasks such as validation, encryption, and data loading. Similarly, in another study [8] an entity called the authenticator evaluates and validates requests. Consequently, in these proposed methods, the EHR manager and the authenticator are vulnerable to attacks.

In study [10], although the patient medical data are stored on the cloud server, only the indices of these data are kept in the blockchain. The study includes a system administrator group consisting of government medical institutions and managers of hospitals. This group is given authority to authorize many actions including user registration, authentication, and system control. The presence of an administrator group in the proposed method may render the system vulnerable to attacks against members of this group. In our method, there is no central entity such as an administrator, EHR manager, authenticator, or system administrator group. The users undertake the work of a central entity by using the consensus mechanism. The absence of a central entity that can be attacked renders our method more robust against attacks.

In studies [9, 15], and [11], patient data are recorded on the blockchain. As the size of the patient data grows, the system becomes inefficient. In addition, as the size of the blockchain increases, scalability decreases, and the SPOF problem increases. In study [12], the encrypted patient EMRs are recorded both on the blockchain and on the cloud server. This is disadvantageous in terms of performance and storage capacity. In study [6], EMR data are recorded on the blockchain. However, there is a calculation performed to
ensure scalability. The proposed scheme also uses an entity called the issuer for authentication, approval, and rejection, and an entity called the verifier to authenticate users and blockchain blocks. In study [7], patient data are kept in a blockchain cloud. In addition, entities called healthcare data gateways are used in the data management layer of the proposed scheme. In our method, patient EMRs are kept in a document-oriented database, whereas the blockchain is used only to save data hashes for access control. This is advantageous for scalability and avoids the SPOF problem.

In the study of [13], medical institutions such as hospitals are held responsible for the execution and verification of smart contracts. In addition, the patient data are kept in different databases in the hospitals. Thus, in case of attacks on the system from both inside and outside the hospitals, the data can be destroyed or access to them blocked. When the data are kept separately in many databases, it creates a disadvantage in terms of integrity and makes data management difficult. In study [5], data are kept in distributed nodes. A blockchain is used to establish connections on nodes. In addition to the encrypted data, the nodes hold large amounts of data, which they call public data, unencrypted. This creates a big problem in terms of security. Moreover, keeping the data distributed creates a disadvantage in terms of integrity and leads to difficulty in managing the data.

4.4 Security Analysis

In this section, various features used in security analysis are examined. Table 3 shows a comparison of the proposed scheme with studies in the literature in terms of these features. P.M. means the Proposed Method in the Table 3, briefly.

<table>
<thead>
<tr>
<th>Properties</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>P.M.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>Confidentially</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>Integrity</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>Availability</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>Avoid Single Point of Failure</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td>Resistance</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>y</td>
</tr>
</tbody>
</table>

4.4.1 Scalability

The complexity increases as the data capacity increases within a system. The system needs to respond quickly to requests to ensure effective performance. Patient EMRs can be large in size. As a blockchain grows in size, its scalability decreases. Therefore, in the proposed method, the data are stored in a document-oriented database instead of a blockchain. The blockchain is used only to store cryptographic digests of data for access control purposes.
4.4.2 Confidentially

The patient EMR is kept encrypted in a document-oriented database. The hash value of the data are kept on the blockchain. In this way, any change on the data can be easily detected and patient confidentiality is protected. Smart contracts are used to prevent malicious access and provide secure access to data.

4.4.3 Integrity

The use of a blockchain in the proposed method prevents data from being modified by undesired manipulations. In addition, smart contracts that leverage the policy list are effective for validating third-party entities. In the proposed method, to prevent unauthorized entities from accessing data and to prevent authorized malicious individuals from destroying or changing data, all entities work with integrity. This is important to strengthen the proposed method against security vulnerabilities and to render it resistant to attacks. In addition, no user is authorized to change or correct the proposed access policy or smart contract agreement.

4.4.4 Availability

In the proposed method, only patients and third-party entities authorized by patients can access patient EMRs through a third-party application. Anyone with access authorization has the opportunity to access the data held on the cloud at any time or place.

4.4.5 Avoid Single Point of Failure (SPOF)

In the proposed method, keeping patient EMRs in a distributed database effectively solves the single point of failure (SPOF) problem. In addition, the blockchain is used only to save data hash values for access control purposes. This situation does not create a problem in accessing the data in the document-oriented database.

4.4.6 Resistance

A central presence in an administrative or managerial position may cause a security problem in a system where medical data are stored and processed. In attacks on a central entity by an attacker, if this entity is compromised, the system becomes unreliable and patient data may fall into the hands of malicious persons. In the proposed method, there is no central entity in the administrator or manager position. All transactions of this entity are carried out through the distributed consensus mechanism.

4.4.7 Assumptions

In our proposed method, EMRs are encrypted using a secure encryption algorithm before being uploaded to the database located on the semi-trustworthy
cloud server. It is assumed that the encryption algorithm used in the method is sufficient for EMRs and that neither internal nor external adversaries can decrypt the encrypted data without using the secret key. The cloud server and malicious users cannot obtain a meaningful message from the encrypted data. The following scenarios were created for the security analysis of the proposed method.

Suppose an attacker gains unauthorized access to data stored in the database.

A patient’s EMR is kept encrypted in a database:

\[ \text{encryptedEMR} = EMR_{(\text{pid}) \text{secretKey}} \]

Even if a patient’s EMR falls into the hands of unauthorized individuals, it is not possible to decrypt the encrypted data without the patient’s secret key \( \text{pid}_{\text{secretKey}} \). An EMR is only meaningful to the patient who owns that data and third-party entities that the patient authorizes.

Suppose an authorized third-party entity maliciously modifies or deletes the patient EMR.

Hash values of EMRs are kept on the blockchain for access control:

\[ \text{Hash}(EMR_{\text{pid}}) \]

These hash values on the blockchain cannot be changed. Therefore, the modification or deletion of EMRs stored in the database by authorized malicious third-party entities can be easily detected. After an authorized third-party entity has completed the analysis of a patient’s EMR, a check is performed as in Algorithm 3. After the analysis, if the hash value of the EMR in the database is the same as the hash value in the blockchain, no change has been made to the data. Otherwise, the data have been changed. In this case, the patient generates a new secret key and encrypts the EMR with its new secret key and uploads it back to the database. The third-party entity \( \text{e}_{\text{id}} \) is removed from the system.

### Algorithm 1 Hash Control

1: Decrypt the \( \text{encryptedEMR} \) by using \( \text{pid}_{\text{secretKey}} \) (by the patient)

\[ \text{decryptedEMR} = \text{encryptedEMR}_{(\text{pid}) \text{secretKey}} \]

2: Hash \( \text{decryptedEMR} \) with a hash function

3: if \( \text{Hash} (\text{decryptedEMR}) == \text{Hash} (EMR_{\text{pid}}) \) then

4: break;

5: else

6: \( \text{revoke(} \text{e}_{\text{id}} \)\)

Generate new patient secret key: \( \text{pid}_{\text{newSecretKey}} \) (by the patient)

encrypt the patient EMR with new secret key

\[ \text{encryptedEMR} = EMR_{(\text{pid}) \text{newSecretKey}} \]

delete previous patient EMR datas and add new datas in database

7: end if

Suppose an attacker sends an unauthorized request message.
Request messages must contain the ID information of the entity requesting authorization. Request messages are checked with a smart contract on the policy list, and an unauthorized request is denied. Steps 2-6 of Algorithm 2 present the control process. Suppose an attacker obtains the ID of a third-party entity and sends a request message. In this case, even if the attacker passes steps 2-6 of Algorithm 2, the request message must be signed with the private key of the third-party entity in order to pass steps 7-11. However, the private key of a third-party entity is kept private only for that individual.

Suppose an attacker is included in the system as a patient.

Patients do not have authority over or access to other patients’ EMRs. Patients can only authorize or deauthorize third-party entities for their own EMRs. The inclusion of an attacker in the system as a patient does not pose a security problem for the EMRs of other patients.

5 Results

In this study, a new method is proposed for the secure storage and rapid sharing of patient EMRs. In the proposed method, a document-oriented database is used to store data and a blockchain is used for access control. The method was examined by evaluating its operating performance and security.

The performance was evaluated using the PoW, the PoS and the PoA consensus mechanisms. The consensus mechanisms were written in Java and the results were obtained by using a computer with i7-7700HQ CPU, a 2.80GHz processor and 16 GB RAM. The consensus mechanisms were compared by block sizes, block creation times and memory usages and the most advantageous in terms of use were discussed. As regards the security evaluation, the proposed method was compared with various aspects of studies in the literature and it was determined to be more advantageous than the other studies.

Declarations

Conflict of interest The author declares that he has no conflict of interest.

Funding The author declares No competing financial and/or non-financial interests and fundings.

Ethics Approval Not applicable.

Consent to publish Not applicable.

Data Availability Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Author Contribution All authors contributed to the study conception and design.

References

An Improved blockchain-based secure medical record sharing method


An Improved blockchain-based secure medical record sharing method


[26] Syed, T.A., Alzahrani, A., Jan, S., Siddiqui, M.S., Nadeem, A., Alghamdi,
An Improved blockchain-based secure medical record sharing method
