A blockchain-based framework for electronic medical records sharing with fine-grained access control

Jin Sun, Lili Ren, Shangping Wang, Xiaomin Yao

School of Science, Xi'an University of Technology, Xi'an, Shaanxi, China

* liren0608@163.com

Abstract

In the medical system, the verification, preservation and synchronization of electronic medical records has always been a difficult problem, and the random dissemination of patient records will bring various risks to patient privacy. Therefore, how to achieve secure data sharing on the basis of ensuring users' personal privacy becomes the key. In recent years, blockchain has been proposed to be a promising solution to achieve data sharing with security and privacy preservation due to its advantages of immutability. So, a distributed electronic medical records searchable scheme was proposed by leveraging blockchain and smart contract technology. Firstly, we perform a hash calculation on the electronic medical data and store the corresponding value on the blockchain to ensure its integrity and authenticity. Then, we encrypt the electronic medical data and store it in the interplanetary file system which is a distributed storage protocol. These operations not only can solve centralized data store of servers of several medical institutions, but also be good at lowering stress from data store and high-frequency access to blockchain. Secondly, the encrypted keyword index information of electronic medical records was stored on the Ethereum blockchain, meanwhile a smart contract deployed in the Ethereum blockchain is used to realize keyword search instead of depending on a centralized third party. Furthermore, we use attribute-based encryption scheme to ensure that only the attributes meeting the access policy can decrypt the encrypted electronic medical records. Finally, our performance analysis and security analysis show that the scheme is secure and efficient.

1. Introduction

In recent years, with the development and application of cloud computing technology, the electronic medical records (EMRs) system has shown a trend of doctor-centered development. In this case, one of the most controversial issue is how to ensure the privacy, security, and sharability of EMRs while achieving fine-grained access control. An effective solution is to combine cloud storage, attribute-based encryption (ABE), and searchable encryption technologies. However, this approach has some challenges in terms of privacy, security, and interoperability.

Firstly, in the health care system, the patient's medical information such as prescriptions, nuclear magnetic resonance image, assay and pathological results are privacy-sensitive and
contain all the information related to personal health. Especially when using various encryption mechanisms to protect personal medical records, and the centralized key management adopted may bring a single point of failure, thereby increasing the risk of data exposure. Second, almost all ABE schemes require a trusted entity authority to distribute attribute keys to the system. It is usually assumed that the entity is fully trusted, but it is difficult to find a fully trusted authority in reality. Moreover, for the EMR outsourcing and storing in the third part, some keywords would be fully utilized while other users seek to make a study and analysis for the required data. However, since EMR involves the privacy of users, privacy preserving should be considered during searching process. Furthermore, the identity authentication of the data requester should be taken into account in the keyword searches to prevent identity forgery of illegal users. Finally, cloud storage often plays an important role in data storage environments. To ensure the data is not tampered with, lost, accessed illegally, or stolen arbitrarily, the most common technique is to encrypt shared content before storing it to a cloud server. However, in practice, the cloud server is always regarded as a semi-trusted entity that can perform a small number of search operations and forge some wrong search results back to the user. These erroneous or incomplete personal medical records may mislead users, (such as doctors, research institutes, or other patients) to make incorrect judgments, which could endanger an individual’s life. Therefore, if we create a shared data source for person such as doctors, caregivers, research institutes, and insurance companies to provide the timely, accurate and extensive data and enabling cross-institutional data sharing, it can help healthcare professionals develop optimal treatment plans. In addition, the secure sharing of EMRs can provide researchers with a broad data set to study disease and to accelerate the development of biomedicine.

Several [1–5] encryption methods have been put forward for these medical data sharing and storage problems, but they are still not enough. Fortunately, a scholar named Satoshi Nakamoto published a new paper [6], “Bitcoin: a peer-to-peer electronic cash system” that expounds the electronic currency theory based on cryptography. After the emergence of Bitcoin, its underlying technology blockchain [7] quickly gained recognition from all walks of life and began to research eagerly. The birth of blockchain technology, with its attractive characteristics such as invariance and decentralization [8, 9] is considered as a powerful fit to provide an appropriate solution to solve these problems and change the centralized storage mode of this medical data. This has attracted widespread attention in academia and industry, including medical [10–12] finance [13] and energy [14–16]. In addition, the blockchain paves the way for building smart contracts [17], which acts as a piece of code on the blockchain and can execute operations once certain criterion be met. More importantly, smart contracts that can be performed autonomously without the need for an honest third party. In addition, the InterPlanetary File System (IPFS) is a global point-to-point distributed version file system. The combination of IPFS and blockchain allows users to process large amounts of data through IPFS, so that the data itself does not need to be placed on the chain, which not only saves the network bandwidth of the blockchain, but also effectively protects it. As for the security of the data, on the one hand, it can be stored in IPFS after encryption; on the other hand, the distributed sharing of the file can be realized by using the IPFS, which effectively solves the problem that the cloud provider deletes part of the data in order to save storage space in the traditional storage mode. Therefore, the combination of blockchain technology and IPFS can provide a good solution for the above problems.

1.1 Our contributions

Based on the above statement, this paper proposed a secure storage and searchable scheme for EMRs with fine-grained access control based on blockchain. Our scheme takes advantage of
the blockchain and smart contract technology to implement a decentralized data sharing, on which it gracefully preserve the authenticity and integrity of EMRs. The main contributions of this article are summarized as follows:

1. A point-to-point distributed storage system IPFS was adopted. The medical data files are encrypted and stored in the IPFS to ensure complete EMRs. This eliminates the needs to put the data itself on the chain, which not only saves the network bandwidth of the blockchain, but also make up for the shortcomings of the existing blockchain system in file storage.

2. Using CP-ABE technology and auditing ideas to achieve doctor-centric access control. That is to say, the new scheme allows doctors and patients to set the policy about who has the right to access EMRs for realizing more secure data management and resisting data forgery. Thus, only data requesters who were authorized and their attributes satisfy the access policy in the ciphertext can decrypt the EMRs.

3. Building an encrypted keyword index for shared medical records, in order to implement EMRs’ secure search in a distributed storage system, keyword index information is stored in a blockchain, and smart contract is deployed on the Ethereum blockchain. Once the smart contract is deployed, it will be executed automatically and operated in good faith.

4. An application example of a medical insurance scenario is given. Finally, safety analysis and performance analysis show that our scheme is practical and feasible.

2. Related work

Quite a few schemes have been proposed to deal with the vulnerability in current EMRs system. In 2010, Ibraimi et al. [18] applied ABE to the security management of personal health records to achieve flexible access control. Nonetheless, a strict safety proof still has not worked out in this paper. In 2013, Li et al. [19] proposed a novel patient-centric framework and a suite of mechanisms for data access control to personal health records stored in semi-trusted servers. And the paper supported multiple data application scenarios. However, since the cloud server is not completely trusted which sometimes tamper with the received encrypted data, data privacy needs to be further studied.

In 2015, XhafaF et al. [20] proposed a cloud-based medical record system which adopts an ABE scheme to encrypt medical information. These data utilized patient’s condition without leaking defined description of illness and the department in which the doctor is located. In fact, the paper adopts a symmetric encryption scheme, but its implementation must depend on an entirely honest Global Authority to take charge of key management, issuing public parameters for the system and generating secret keys for the doctor, that is one of the main shortcomings of this scheme.

A mobile application for medical data sharing was implemented in work [21], but it was limited to patients and doctors. In addition, the scheme did not support without supporting data sharing with third parties such as insurance companies and medical research centers. Similarly, reference [8] proposed interoperability proof to avoid computational cost. This method uses a single centralized source of trust for transactions to support network consensus. However, the literature fails to achieve fine-grained access control. In 2017, Q. Xia et al. [22] put forward a blockchain-based health data sharing framework that fully solves the access policy challenges related with sensitive information stored in the cloud. The framework is based on a licensed blockchain that only allows the authorized and the invited users to access. In addition, in order to offer a source of medical data, auditing and secure data tracking, the
authors used access control and smart contracts mechanisms in their other paper [10] to effectively track data action and to revoke access to violating entities, but none of them achieve the searchability of data. Blockchain is used in [23] to ensure that users have reliable access to electronic health records. Further, the author relies on smart contracts to manage physicians’ electronic medical record usage and focus on theoretical analysis, so the feasibility of the proposed solution has not been confirmed in the actual electronic health records sharing scenario. In 2018, Zhang et al. [11] proposed a data sharing scheme for implementing privacy protection in personal health information (PHI) system through the consortium blockchain. The scheme constructed two kinds of blockchain by designing data structures and consensus mechanisms. On the basis of blockchain, in order to achieve secure search, privacy protection, data security and access control, PHI sharing protocol is proposed by using public key encryption and keyword search technology. However, the scheme does not achieve complete control over the data, such as data revocation, setting data access deadlines, and so on. In 2018, Sun et al. [24] proposed a searchable personal health record framework with fine-grained access control in cloud-fog computing, which adopted CP-ABE to outsource part of the computational operations to fog nodes, effectively reducing the computational burden in cloud storage. However, the implementation of this solution must rely on multiple authorization centers, so the confidentiality of the data cannot be guaranteed. At the same year, Zhang et al. [25] proposed a FHIRChain based on blockchain architecture for sharing clinical data, and demonstrated a distributed application based on FHIRChain, which uses digital health identity to authenticate participants. However, the performance of the FHIRChain prototype is not strict enough, and fine-grained access control is not implemented for the identity management of the group.

In 2019, Kurt et al. [26] offered some ways to modern medical infrastructure by taking advantage of the potential value of the blockchain technology. However, these approaches are only a theoretical explanation and there are no related schemes or specific implementations. In the same year, DINH [27] et al. designed a reliable access control mechanism to manage user access based on a single smart contract to ensure efficient and secure electronic health record sharing, but the cryptographic algorithms involved in data protection were less secure.

3. Preliminaries

Next, we give some background knowledge used in this article, such as blockchain, smart contract, IPFS, and some cryptographic tools throughout the whole paper.

3.1 Blockchain

In 2008, Satoshi Nakamoto first proposed the concept of blockchain [6], it is called Bitcoin as the underline technology behind the cryptocurrency, which helps to realize peer-to-peer value exchange without centralized third parties. The process about storage, accounting, maintenance, verification and transmission of blockchain data is based on the distributed system structure, using pure mathematical methods rather than central institutions to establish trust relationships among distributed nodes, thus forming a decentralized trusted distributed system. Fundamentally, it is a set of nodes named verifiers or miners that are responsible for maintaining a trusted record for all transactions via a consensus algorithm in a trust-free environment.

As the name implies, a blockchain is made up of many blocks, among which the block refers to the collection of all information communication data in the system within a period of time. A block is the basic unit that forms blockchain, and each block has a timestamp as its unique mark to ensure the traceability of the blockchain. Each block is divided into two parts: Block Header and Block Body, each block header contains the link pointers of the block headers of
the previous block, the Merkle root of the tree-like transaction information, and a timestamp. And the block body records the data information in the network. In this way, the blocks are linked together in chronological order. The blockchain structure is shown in Fig 1.

3.2 Smart contract

A smart contract is a special account with associated data and code. The code is in an Ethereum-specific binary format (i.e., Ethereum Virtual Machine byte code) and deployed by an account to a global database known as blockchain. Smart contracts are an exciting application of blockchain. The smart contract of the blockchain can be understood as a piece of code (contract) that triggers execution when the two parties make a transaction transfer on the blockchain asset. The operation mode is shown in Fig 2. A smart contract usually provides many functions or application binary interfaces (ABIs) that can be used to interact with it. These ABIs can be executed by sending a transaction from an account or a message to another contract. In this paper, the doctor creates a smart contract for storing the keyword index of the patient’s medical records and some related data, in the meantime guaranteeing a fair search. In the absence of a trusted third-party entity, data requester mortgages search fees to smart contracts, and smart contracts help users retrieve data. If the wrong search data is returned, the service fee cannot be charged.

3.3 IPFS

IPFS (InterPlanetary File System) [28] is a peer-to-peer data distribution protocol in which nodes came form a distributed file system. When we upload the file to the IPFS network, the
system will split the file into N fragments, each fragment generates a hash value, and all the fragment hashes are combined to generate a hash value of the total file. This hash value is the address of the file within the IPFS. When querying a hash to IPFS, IPFS can quickly find the node owning the data and retrieve it by using a distributed hash table, and then use the hash to verify whether it is the correct data. The IPFS network is a fine-grained, distributed and easily federated content distribution network that is common to all data types including images, video streams, distributed databases, operating systems etc, and has fewer storage restrictions.

3.4 Cryptographic tools

The main symbols used in the remaining articles are showed in Table 1.

3.4.1 Attribute-based encryption. Attribute-based encryption introduced by Sahai and Waters [29] is considered an expansion of identity-based encryption [30] by treating identity as a set of attributes. So far, there are two types of ABE schemes, namely ciphertext policy attribute-based encryption (CP-ABE) and key policy attribute-based encryption (KP-ABE), which are introduced by Goyal et al [31]. In our scheme, we adopt CP-ABE scheme proposed in to enable the doctor and patient to identify the data requester who have the attribute to decrypt an EMR.

3.4.2 Symmetric searchable encryption [32]. Symmetric searchable encryption involves three entities, data owner, server and data user. The data owner and the data user can be the
returns the related search result. The smart contract first determines the identity of the data requester and then medical record, he uses his own search private key to generate a search token to call the smart contract. When the data requester needs to search a matched documents to the user. In our scheme, the doctor generates a keyword index of the medical identity with the received search token and index, and then returns the successfully send the search token to the cloud server. The cloud server retrieves the corresponding documents to the keyword according to his private key, and then sends them to the cloud server. When the data user needs to search for a keyword cloud server, he encrypts these documents to generate ciphertext and the corresponding index, same person. Assume that the data owner has documents that need to be stored on the private cloud server, he encrypts these documents to generate ciphertext and the corresponding index, and then sends them to the cloud server. When the data user needs to search for a keyword that he interested, he generates a search token of the keyword according to his private key, and send the search token to the cloud server. The cloud server retrieves the corresponding document identity with the received search token and index, and then returns the successfully matched documents to the user. In our scheme, the doctor generates a keyword index of the medical record and stores it in the smart contract. When the data requester needs to search a medical record, he uses his own search private key to generate a search token to call the smart contract. The smart contract first determines the identity of the data requester and then returns the related search result.

3.4.3 Bilinear map [25]. Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative cyclic groups of prime order $p$. Let $g_0$ denote one generator of $\mathbb{G}_0$. A bilinear pairing operation constructed as $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ with the following properties:

1. Bilinearity: For $\forall u, v \in \mathbb{G}_0$ and all $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$ holds.
2. Non-degeneracy: $e(g_0, g_0) \neq 1$.
3. Computability: For all $\forall u, v \in \mathbb{G}_0$, $e(u, v)$ can be efficiently computed.

3.4.4 General bilinear group [33]. Let $\phi_0, \phi_1$ be two mapping functions $\mathbb{Z}_p^k \rightarrow \{0, 1\}^k$, where $\mathbb{Z}_p^k$ is an additive group, $k > 3 \log(p)$. The groups $\mathbb{G}_0, \mathbb{G}_1$ are defined as $\mathbb{G}_0 = \{\phi_0(\chi) | \chi \in \mathbb{Z}_p\}$, $\mathbb{G}_1 = \{\phi_1(\chi) | \chi \in \mathbb{Z}_p\}$ respectively, where $\mathbb{G}_0$ is a general bilinear group. $g$ denotes $\phi_0(1)$, and $g^x$ represents $\phi_0(x)$, $e(g, g)$ denotes $\phi_1(x)$ and $e(g, g)^x$ represents $\phi_1(x)$.

3.4.5 Access structure. There are several kinds of access structure in ABE scheme, such as AND-gate structure, tree-based structure and threshold structure. In our protocol, we adopt a series of AND gate as our access policy.

Let assume the total number of attributes is $n$, and all attributes be indexed as $U = \{x_1, x_2, \cdots, x_n\}$. For every attribute $x_i \in U$, $(i = 1, 2, \cdots, n)$, $V_i = \{v_{i,1}, v_{i,2}, \cdots, v_{i,m}\}$ be a set of possible values of $x_i$, where $n$ is the number of the possible values for $x_i$. Then the attribute list $S$ for a user is $S = (att_1, att_2, \cdots, att_m)$, where $att_i \in V_i$. The access policy in ciphertext is $A_s = (A_{s_1}, A_{s_2}, \cdots, A_{s_n})$.

Table 1. Symbols table.

| Symbol | Descriptions |
|--------|--------------|
| $U$    | Attribute universe |
| $\text{MSK}$ | Master key |
| $PK$ | Public key |
| $uk$ | Attributes private key |
| $m$ | The original medical records |
| $pk_{ij}$ | Public-private key pair of the patient |
| $pk_{ij}$ | Public-private key pair of the doctor |
| $AV_{ij}$ | The medical records authentication value |
| $h_{\text{location}}$ | The encrypt medical records' location |
| $W_m$ | Keywords set |
| $\text{index}$ | The index of the keywords |
| $Tw$ | Search token |

https://doi.org/10.1371/journal.pone.0239946.t001
where \( \mathcal{A}_i \in \mathcal{V} \). We define the attribute list \( \mathcal{S} \) satisfies the access policy \( \mathcal{A} \) if and only if \( \text{att}_i = \mathcal{A}_i \), \( (i = 1, 2, \ldots, n) \).

3.4.6 Security model. We designed security game to prove the security of our scheme, which depends on the general bilinear group model and the game is semantically secure against an adaptive chosen plaintext attack (IND-CPA).

**IND-CPA game:**

**Initialization:** The adversary \( \mathcal{A} \) submits an access structure \( \mathcal{A}^* \) that he wishes to challenge it.

**Setup:** The challenger \( \mathcal{B} \) runs the global setup algorithm to generate the public key \( \text{PK} \) and master key \( \text{MSK} \), then sends the public key \( \text{PK} \) to adversary \( \mathcal{A} \), and holds the master key \( \text{MSK} \) privately.

**Phase1:** \( \mathcal{A} \) can repeatedly ask the \( \mathcal{O}_{\text{SK}}(\mathcal{S}) \) oracle.

\( \mathcal{O}_{\text{SK}}(\mathcal{S}) \): \( \mathcal{A} \) can issue queries for private key \( \text{SK} \) by submitting attribute set \( \mathcal{S} \).

**Challenge:** \( \mathcal{A} \) submit two equal-length messages \( m_0 \) and \( m_1 \), \( \mathcal{B} \) chooses a random bit \( b \in \{0, 1\} \) and encrypts \( m_b \) under \( \mathcal{A}^* \), then send the challenge ciphertext \( \text{CT} \) to \( \mathcal{A} \).

**Phase2:** The same as **Phase1**, with the restriction that \( \mathcal{S} \) does not satisfy \( \mathcal{A}^* \).

**Guess:** \( \mathcal{A} \) outputs a guess bit \( b' \in \{0, 1\} \). If \( b' = b \), he wins the game, otherwise, he fails.

Therefore, the \( \mathcal{A} \)'s advantage is defined to be \( \text{Adv}_{\mathcal{A}} = |\Pr[b' = b] - \frac{1}{2}| \).

4. System framework

In this section, we briefly describe the system framework of our scheme. As shown in Fig 3, the system mainly includes five entities: patient, doctor, data requester, IPFS, and blockchain network. Possible interactions between different entities in the paper are shown in Fig 3. Below we discuss the roles of each entity in the system and their interactions.

4.1 Roles of each entity

**Patient.** Patient is also a data user, suppose a patient \( i \in \mathcal{U} \) registers with the hospital \( k \in \mathcal{H} \) to see a doctor \( j \in \mathcal{D} \). The server of the hospital \( k \) generates a certificate \( \varphi_i \) for the patient and returns it to the patient. At the same time, the server stores \( \mu_i = H(\varphi_i) \) on the service registration list of the doctor \( j \). Here \( \varphi_i \) is equivalent to a number plate for the doctor to generate medical information for patient \( i \). When the patient goes to the doctor, he will give the doctor a certificate \( \varphi_i \) to generate his personal medical information and access his historical medical records.

**Doctor.** The doctor is responsible for generating medical information for the patient and extracting a set of keyword for the patient’s medical information after obtaining the patient’s consent. In the scheme, doctor is assumed to be honest and not to collude with data requester to get patients’ medical records for illegal benefit. Furthermore, in order to safely store interoperable data, the patient and the doctor discuss an access policy to encrypt the medical information and the keyword to generate the ciphertext. The ciphertext is composed of two parts, one is the ciphertext of the patient’s medical information that is stored on the IPFS, and the other part is the ciphertext of the keyword, which is stored on the smart contract.

**Data requester.** The data requester is the data user, such as scientific research institutions, medical insurance companies, patients’ families, etc., which can obtain relevant records as long as their attributes meet the corresponding access policy.
**IPFS.** The IPFS stores the ciphertext of the patient’s medical information encrypted by the doctor and then returns the corresponding hash value to the doctor, where the hash value is equivalent to a file address. Once the address is obtained, the authorized data requester can download the ciphertext according to the address.

**Blockchain network.** The blockchain network is the core of our paper, and it performs smart contracts in a distributed manner without relying on central entities, which is necessary to ensure the secure storage and sharing of electronic medical information. In order to improve system efficiency, we use the Ethereum blockchain. In our scheme, we do not consider the miners on the Ethereum blockchain for the time being.
4.2 The interaction process in Fig 3

Fig 3 is a major system model of our scheme. Below we briefly analyze each step in the model.

① The patient goes to the hospital to see a doctor and interacts with the doctor to generate his medical data.

② The doctor deploys a smart contract on the blockchain, and then generates a hash value of the signed medical data and a message authentication value stored on the blockchain.

③ The doctor encrypts the medical record under the access policy negotiated with the patient and uploads it to IPFS.

④ IPFS returns the location to the doctor where the medical records are stored.

⑤ The doctor encrypts the medical record location, then embeds the ciphertext into the transaction and broadcasts the transaction to the blockchain.

⑥ When the transaction is confirmed, the doctor records the transaction address.

⑦ The doctor encrypts the keywords of the medical records, generates a keyword index, and then stores the transaction address and keyword index into the smart contract.

⑧ The data requester sends a medical record access request to the doctor, and the doctor authenticates the identity of the data requester and adds it to the authorized user of the smart contract.

⑨ The doctor distributes the appropriate attributes as well as generates his attribute key, which is then returned to the data requester through a secure channel.

⑩ The data requester creates a search token and then invokes the smart contract with the token as a parameter.

⑪ The smart contract judges the identity of the data requester. If he is an authorized user, then smart contract returns the relevant search results to him.

⑫ The data requester reads the transaction information from the blockchain and calculates the file location.

⑬ The data requester obtains the file location and then downloads the encrypted medical record from the IPFS.

⑭ The data requester judges whether his attributes satisfy the access structure in the ciphertext, and if so, decrypts and obtains the medical record.

4.3 Protocol description

A sketch of the possible interactions between the different system entities is shown in Fig 4. Our proposed framework consists of three parts: system establishment, medical data generation and storage, data search and access. Each part in the system is discussed in the following paragraphs.

Step 1: System establishment. In the first place, the doctor generates the system public key PK by implementing the function GlobalSetup(λ), where λ is the safety parameter. And each patient and doctor generates their own public and private keys by manipulating function KeyGen(PK). It including three sub-algorithms:

- GlobalSetup(λ): Let $U$ be the attribute universe, $g_0$ be the generator of $\mathbb{G}_0$, and bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$. The doctor selects two random numbers $\alpha, \beta \in \mathbb{Z}_p^*$ and a hash function
H : \{0, 1\}^\ast \rightarrow \mathbb{G}_0$, calculates $A = e(g_0, \xi)^{\mu}$, and then defines a pseudo-random function $F:\{0,1\}^\ast \rightarrow \{0,1\}^\ast$. The public key is published as $PK = (\mathbb{G}_0, \mathbb{G}_1, g_0, e, H, A, B, F)$, the master key is $MSK = (\alpha, \beta)$, and then randomly selects $\xi \in \mathbb{Z}_p^\ast$ and sets the search private key $sk_{\xi} = \{\xi\}$. Hereafter, the doctor deploys a smart contract on the Ethereum blockchain, which is used to store the encrypted keyword index and provide search services for the data requester. After the contract is successfully deployed, the doctor records the contract’s account address and the contract’s ABI.

- **KeyGen\(_i\)(PK):** Each patient $i \in \mathcal{U}$ randomly selects the number $y_i \in \mathbb{Z}_p^\ast$ as his private key $sk_{y_i}$ and then outputs the public key $pk_{y_i} = g_0^{y_i}$.

- **KeyGen\(_j\)(PK):** Each doctor $j \in \mathcal{D}$ randomly selects the number $x_j \in \mathbb{Z}_p^\ast$ as his private key $sk_{x_j}$, and then outputs the public key $pk_{x_j} = g_0^{x_j}$.

**Step 2: Medical data generation and storage.** When a patient $i$ registers with the hospital $k$, the hospital’s server randomly selects $\varphi_i \in \mathbb{Z}_p^\ast$ and sends $\varphi_i$ securely to patient $i$. At the same time, the server selects a doctor $j$ for the patient (the doctor can also choose the patient himself, which is equivalent to the registration in our current medical system), then the server calculates $\mu_i = H(\varphi_i)$. When the patient goes to see the doctor, he presents $\varphi_i$ as his evidence and the doctor generates his medical records $m$. After patient $i$ interacts with doctor $j$, the doctor
would do several steps as follow: (1) extracts the keywords from the medical records to form a set of keywords $W_m$; and then the doctor negotiates an access policy with the patient for the medical record $m$; (2) generates data $Data = \{H(m); Sig_p[H(m)]\}$ that needs to be saved on the blockchain, where symbol $H(m)$ represents hash the medical record $m$, symbol $sig_p[H(m)]$ represents the hash value of medical records signed by the doctor and the patient using their private keys $sk_i$ and $sk_j$ respectively; (3) generates a message authentication value $AV_i = (pk_i, pk_j, P)$, where $pk_i$ and $pk_j$ represent the public key of the patient and the doctor, respectively, and $P$ represents a negotiated access policy; (4) creates a data store transaction, then broadcast data $Data = \{H(m); Sig_p[H(m)]\}$ and authentication value $AV_i = (pk_i, pk_j, P)$ to the blockchain network.

Note: This signature is signed by the doctor and patient commonly to ensure the accuracy and authenticity of the medical record. That is to say, to prevent the doctor tampers with the medical record before encrypting it, a data requester can obtain $Data$ and $AV_i$ from the blockchain network to verify the integrity of the medical record when he obtains the medical record.

Finally, the doctor encrypts the patient’s original records $m$ and keyword sets $W_m$ using the following two sub-algorithms based on the access policy negotiated with the patient.

- **Record-Encrypt($K$):** The encryption algorithm selects a random number $s \in \mathbb{Z}_p$ and sets $C = m \cdot A' \cdot C'$, $\tilde{C} = B' \cdot \tilde{C}$, $\hat{C} = g_{\hat{c}}$. The access policy negotiated between the doctor and the patient is AND gate $\mathsf{A}_i = (\mathsf{A}_{i1}, \mathsf{A}_{i2}, \ldots, \mathsf{A}_{in})$, for each attribute in the AND gate $\mathsf{A}_i$, sets $C_i = H(A_i)^i$, $(i = 1, \ldots, n)$. The ciphertext is $CT = (\tilde{C}, \tilde{C}, C, C)$, then the doctor uploads the $CT$ to IPFS, and records the file location $h_{\text{location}}$ returned by IPFS. In addition, to encrypt the file location $h_{\text{location}}$ the doctor randomly selects the AES key $K_i$, calculates $CT^r = Enc_K(h_{\text{location}})$, then embeds $CT^r$ into the transaction $TX_{\text{ct}}$, and broadcasts $TX_{\text{ct}}$ to the blockchain. When the transaction $TX_{\text{ct}}$ is confirmed, the transaction $txid$ and the corresponding encryption key $K_i$ are recorded.

- **Index-Gen:** The keyword index generation algorithm is run by the doctor. For each keyword $w_i \in W_m$ extracted from the medical record $m$, the doctor calculates $h_i = F(sk_m1||w_i)$, $n_i = F(sk_m2||w_i)$, $txid = n_i \oplus txid$, $K_i = n_i \oplus K$, and then stores the keyword index $\text{index} = (h_i, txid, K_i)$ to the smart contract.

**Step 3: Medical data search and access.** If the patient goes to see the doctor again, he can show $q_i$ directly, and the doctor will treat him by calculating $\mu_i = H(q_i)$. If the patient is going to see a doctor across a hospital, or a research institution wants to obtain the medical data for research, or other legitimate data requester wants to access the medical data, they can obtain the medical data through the following four sub-algorithms:

- **Key-Generation:** When a data requester needs to obtain medical record of the patient, he first sends an access request as well as submits his Ethereum account public key to the doctor. The doctor authenticates the identity of the data requester and distributes the appropriate attributes $S = \{att_{i1}, att_{i2}, \ldots, att_{in}\}$ to him, then adds his account address to the authorized user set of the smart contract. Besides, the key generation algorithm generates attribute secret key and search secret key. The algorithm randomly selects elements $r_i \in \mathbb{Z}_p (i = 1, 2, \ldots, n)$, sets $r = \sum_{i=1}^{n} r_i$, $\tilde{D} = g_{\tilde{d}}^r$. For each attribute $att_{ij} \in S$, randomly choose $r' \in \mathbb{Z}_p$, let $D_{ij} = g_{\tilde{d}}^{r'} \cdot H(\text{att}_{ij})^r$, $\tilde{D} = g_{\tilde{d}}^r$. The attribute secret key and search secret key of data requester are defined as $sk = (\tilde{D}, D_{ij} | i = 1, 2, \ldots, n)$ and $sk = \xi$ respectively, where the search secret key is same for different data requester. The doctor then sends the secret key $\text{SK} = (uk, sk)$ and the address of the smart contract to the data requester via a secure channel.
pany to pay for medical costs when conducting audit claims. In fact, there are some problems in
The doctor encrypts the medical record based on the negotiated access structure, writes
②③
④⑤⑥⑦
①
 entities: insurance company, patient and hospital. The specific workflow is shown in Fig 5.
provided from our framework. Take medical insurance as an example, the process involves three
illegally received the claim. Based on the above issues, the completeness, timeliness, and espe-
of medical records. For example, after the patient and some members of the hospital reached an
traditional medical record management methods and some understandings in the authenticity
process, there are lots of medical insurance cases that cheat the insurance companies to make
has increased, and insurance claims have also increased. However, in the real insurance claims
4.4 Application of our scheme: Insurance claims
In recent years, with the development of medical insurance business, the number of participants
increased, and insurance claims have also increased. However, in the real insurance claims
process, there are lots of medical insurance cases that cheat the insurance companies to make
compensation by falsifying medical records. The medical record is a document that truly rec-
ords the patient’s health information; it is also an significant evidence for the insurance com-
pany to access the data.

• TokenGen(\(\tilde{w}, sk\)): The algorithm is run by the data requester, he creates a search token for
his chosen keyword \(\tilde{w}\) from the medical record and his search secret key \(sk\) as:
\(T\tilde{w} = F(sk, 2||\tilde{w})\). Hereafter, he uses the token as a parameter to invoke the smart contract.

• Test: The test algorithm is run by the smart contract. Firstly, the smart contract judges
whether the data requester is an authorized user. If not, the search operation is refused, other-
wise the search result \(\text{result} = (S_k, S_{\text{txid}})\) is returned according to the token, where \(S_k\) is the
 corresponding encrypted \(K\) set and \(S_{\text{txid}}\) is the encrypted \(txid\) set.

• Decrypt: The data requester obtained the search result from the smart contract, he cal-
culates \(n = F(sk, 2||\tilde{w})\). For \(txid_i \in S_{\text{txid}}\), \(K_j \in S_k\), he calculates \(txid_j = n \oplus txid_i\), \(K_j = n \oplus K_i\).
Then data requester reads the transaction data \(txid_i\) from the Ethereum blockchain and
calculates \(h_{\text{location}} = Dec_{\psi_{K_j}}(CT^i)\), where \(Dec_{\psi_{K_j}}(CT^i)\) indicates that the AES algorithm
is used to decrypt \(CT^i\) and the decryption key is \(K_j\). The data requester decrypts and obtains
\(h_{\text{location}}\), downloads the encrypted medical record from the IPFS, and then to judge
whether the attribute satisfies the access policy, that is \(att_i = h_{\psi}(i = 1, 2, \cdots n)\). If not sat-
ished, he returns to terminate and reads the next transaction data. Otherwise, the medical
record \(m\) is restored as
\(E = \prod_{i=1}^{n} \left( e\left(D_i, C_i\right) / e\left(D, C\right) \right)^{\alpha_i} \left( e\left(g_0, g_0^\alpha\right)^{\alpha_i} \right)^{\alpha_i}\), the algorithm is decrypted by calculation
\(m = C / (e(D, C) / E)\).

4.4 Application of our scheme: Insurance claims
In recent years, with the development of medical insurance business, the number of participants
has increased, and insurance claims have also increased. However, in the real insurance claims
process, there are lots of medical insurance cases that cheat the insurance companies to make
compensation by falsifying medical records. The medical record is a document that truly rec-
ords the patient’s health information; it is also an significant evidence for the insurance com-
pany to pay for medical costs when conducting audit claims. In fact, there are some problems in
traditional medical record management methods and some understandings in the authenticity
of medical records. For example, after the patient and some members of the hospital reached an
agreement, the doctor or the hospital alone forged the patient’s medical information and then
illegally received the claim. Based on the above issues, the completeness, timeliness, and espe-
cially authenticity of medical records must be ensured. What’s more, a proper solution is pro-
vided from our framework. Take medical insurance as an example, the process involves three
entities: insurance company, patient and hospital. The specific workflow is shown in Fig 5.

① The patient negotiates with the doctor about the medical record access structure, including
allowing the insurance company to access the data.

②③ The doctor encrypts the medical record based on the negotiated access structure, writes
the relevant description of the medical record into the blockchain, and stores the encrypted
complete initial medical record on the IPFS.

④⑤⑥⑦ The patient filed a claim with the insurance company. After receiving the applica-
tion, the insurance company will conduct identity authentication to the doctor to obtain
the relevant private key, and then generate the search token with the corresponding key-
words to invoke the smart contract, so as to obtain the address information of the medical
record in the IPFS from the blockchain. Finally, the corresponding encrypted medical
record can be downloaded and decrypted using its secret key.
Insurance company interacts with the blockchain to verify the integrity and authenticity of the decrypted data. If the medical record is true and accurate, the insurance company completes the claim.

Fig 5. Insurance claims.

https://doi.org/10.1371/journal.pone.0239946.g005
4.5 Smart contract creation

The doctor deploys a smart contract by broadcasting a transaction to the blockchain network. The deployed contract is described in the algorithm 1 and contains the following principal features:

**Add authorized users.** When a data requester needs to obtain medical record of the patient, he first sends an access request as well as submits his Ethereum account public key to the doctor. The doctor certifies the identity of the data requester and then adds his account to the smart contract of the authorized.

**Search medical records.** The keyword index of the medical record is stored on the smart contract. And the data requester generates a search token to invoke the smart contract, then the contract first judges whether the data requester is an authorized user, and if so, returns the relevant seek result; otherwise, the operation is terminated. This operation can avoid the drawbacks of data storage on the cloud server. In other words, the cloud server is semi-honest and curious, he will return incorrect or partial search results to the data requester during the search process.

Algorithm 1: Pseudocode for the EMRs Sharing contract

```solidity
contract EMRs Sharing //Doctor

address public Doctor;

mapping (address => bool) public authorizeRequesters;
// requester authorize

mapping (bytes => indexSet[]) Index;
// storage index

event Searchevent (bytes key, bytes result);

function addRequester (address newRequester) onlyDoctor() public
if (!authorizeRequesters[newRequester])
    authorizeRequesters[newRequester] = true;
end
return authorizeRequesters[newRequester];

function addIndex (keywordIndex, txid, key) public
if emr.sender is not doctor then
    throw
end
mapping keywordIndex to (txid, key), and add it to index variable collection;
return True

function search(Token) haveAuthorized(Requesters) public
if tx.origin is not Doctor then
    throw
end
if Requester not in authorizeRequesters list then
    throw.
end
length = Index[Token].length;
result = new strings.slice[](length);
result1 = new strings.slice[](length);
for i = 0; i < length; i++ do
    result[i] = (Index[keywordHash][i].key);
    result1[i] = (Index[keywordHash][i].txid);
end
Searchevent(emr.sender, result, result1);
return (result, result1)
```

```
5. Security

5.1 Security analysis

This part conducts a security analysis on the proposed scheme from three aspects: secure storage, privacy protection and tamper-proof.

Secure storage. The storage security of data is an important feature of this paper. The whole process of production and use of the EMRs is secure in the scheme. Public information on medical records is stored on the blockchain and cannot be tampered with and publicly visible. As the medical record producer, the doctor performs a hash operation on the medical record and stores it on the blockchain. The original medical records are then encrypted and stored in the IPFS and the location returned by the IPFS is encrypted and written into the blockchain, ensuring the authenticity and confidentiality of the data source. Furthermore, the distributed storage characteristics of IPFS guarantee the security of the medical records storage.

Privacy protection. In the first place, the data requester participate in the transaction on the blockchain with an anonymous way, and each transaction can output different public-private key pairs, which effectively protects the identity information of the data requester to a certain extent. Secondly, the public hyperledger does not contain the patient’s privacy information, and the complete medical record is encrypted and stored in the IPFS. The producer of the medical data maybe retain the raw data of the patient’s medical record, which requires related mechanisms and laws to prevent doctors and medical institutions from revealing the user’s medical record to protect the patient’s privacy. Thirdly, only the address information of the ciphertext is stored in the blockchain, and the unauthorized data requester cannot obtain the corresponding location. In addition, if the attributes of data requester don’t satisfy the access policy embedded in the ciphertext, and the plaintext information of the medical record cannot be obtained, so it is impossible to get any actual data about the medical record from the public information of the blockchain.

Tamper-proofing. All information on the blockchain is public, tamper-proof and in a chronological order. The distributed consensus mechanism on the blockchain makes its trust based on cryptographic algorithms without relying on trusted third provider. Once the data is written into the blockchain, it cannot be tampered with, because each block is saving the hash value of its previous block. If one want to modify the data of a block, needing at least 51% of the total calculating power, which is almost impossible. The hash value of the original data of the medical record is preserved in the blockchain, and any change of the original data will cause a change of its hash value, so it also directly guarantees the inelastic non-tampering of the medical record.

5.2 Security proof

Theorem 1. In the general group model, for any adversary $A$, let $q$ be a bound on the sum number of group elements it receives from queries it makes to the oracles for the hash function, groups $G_0$ and $G_1$, and the map $e$, and from its interaction with the IND-CPA security game. Then we have that the advantage of the adversary in the IND-CPA security game is $O(q^2/p)$.

Proof. The initialize $g = \phi_0(1), g^e = \phi_0(x)$, that is to write $e(g,g) = \phi_1(1), e(g,g)^x = \phi_1(x)$. The adversary $A$ will interact with the challenger $B$ using the $\phi$- denotes of the group elements. In the security game $B$ communicates with $A$ as follows:

Setup: Challenger $B$ randomly selects $\alpha, \beta \in Z_p^*$, and calculates the public key is $PK = (A = e(g_0, g)^\alpha, B = g_0^\beta)$, the master key is $MSK = (\alpha, \beta)$, then sends the public key $PK$ to the adversary $A$, and keeps the master key $MSK$ privately.
When the adversary (or challenger) invokes for the evaluation of $H$ on any attribute $att$, a new random number $t_i$ is selected from $Z_p^*$, and the simulation provides $g_i^{\theta}$ as the response to $H(att)$.

**Phase1:** $A$ sends a query to the $O_{\alpha}(S)$ oracle:

$O_{\alpha}(S)$: The adversary $A$ issues attribute private key query to the set $S = \{att_1, att_2, \ldots, att_n\}$. The challenger $B$ selects a new random value $r \in Z_p^*$, calculates $d = g_0^{\alpha \cdot r}$, and for each attribute in $att_i \in S$, $i = (1, \ldots, n)$, it selects new random numbers $r_s, r'_s$ from $Z_p^*$, sets $D_i = g_i^{\alpha \cdot r_s}, \tilde{D} = g_i^{r'_s}$. The attribute private key is defined as $uk_{\alpha} = \{D, \tilde{D}, d\}$. Then sends the private key to adversary $A$.

**Challenge:** $A$ generates a pair of messages $m_0$ and $m_1$, hoping to challenge it. In addition, $A$ announced a challenge access structure $A$. The challenger $B$ creates a challenge ciphertext, he first selects a random number $s \in Z_p^*$ and computes $C = m_0 \cdot e(g_0^s)^{\alpha \cdot s}, \tilde{C} = g_i^s, C = g_i^s$. Hereafter, for each attribute in the AND gate, $B$ sets $C_i = g_i^{s'_i}, i = (1,2,\ldots,n)$, and then these values are sent to $A$.

**Phase2:** The same as **Phase1**, with the restriction that $\tilde{S}$ does not satisfy $A$.

**Guess:** $A$ returns a guess $b \in \{0,1\}$.

We can consider an improved game where the truly challenging ciphertext is replaced by $e\left(g,g\right)^{\theta}$ instead of $m_0 \cdot e(g,g)^{\alpha \cdot s}$, and the probability of distinguishing $m_0 \cdot e(g,g)^{\alpha \cdot s}$ from $e\left(g,g\right)^{\theta}$ is equal to half the probability of distinguishing $m_1 \cdot e(g,g)^{\alpha \cdot s}$ from $m_0 \cdot e(g,g)^{\alpha \cdot s}$.

Next, a detailed shows of the $B$’s simulation is depicted. If there is no “accidental collision” that the $B$’s simulation is perfect. More specifically, we think of an oracle query as being a rational function $\delta = \eta/\xi$ in the variables $\theta, \alpha, \beta, r, r', s, s'$. An unexpected collision would be when two queries corresponding to two distinct rational functions $\delta = \eta/\xi, \delta = \eta'/\xi'$, but where since the random selects of these variables’ values, we have that the values of $\delta = \eta/\xi = \eta'/\xi' = \delta'$, where $\eta \neq \eta', \xi \neq \xi'$.

This unexpected collision does not occur in group $G_0$ or $G_1$, for our new condition. For any pair of queries corresponding to different rational functions $\eta/\xi$ and $\eta'/\xi'$, a unexpected collision only occurs when there is a non-zero multiple $\eta \xi - \xi' \eta' = 0$. According to the literatures [34], this event’s probability is $O(1/p)$. And then in the constraint conditions, the probability of accidental collision happens is at most $O(\xi'^2/p)$.

Now we state that even if $B$ replaces $as$ with the variable $\theta$, the $A$’s view would have been identically distributed. Due to $\theta$ only appears as $e\left(g,g\right)^{\theta}$, if a collision occurs, we have $\delta - \delta' = \gamma \alpha s - \gamma \theta$, where $\gamma \neq 0$ is a constant. Then we can artificially add query $\delta - \delta' + \gamma \beta = \gamma \alpha s + \Sigma \gamma' s r$. In order to get an inquiry from $\gamma \alpha s$, the adversary $A$ does not need to in the item of the form $\Sigma \gamma' s r$. However, according to the simulation, the adversary $A$ cannot get the master key. Therefore, the adversary $A$ cannot get the form $as$ and construct a query for $e\left(g,g\right)^{\gamma \alpha s}$.

## 6. Performance evaluation

In this section, we provide performance evaluation of the proposed scheme. Firstly, we compare the security properties among the proposed protocol and several other literatures. Later, the computational overhead of the cryptographic operations are analyzed. Finally, we evaluated the performance in the cost of smart contract and the system data throughput.

### 6.1 Comparison of schemes characteristics

We have chosen the recently proposed medical record sharing schemes [5, 8, 10, 12, 25], as a benchmark. Table 2 compares some of the features of the blockchain-based protocols Peterson.
Xia [10], and non-blockchain based protocols Au [5], Yang [12], and Sun [25]. As it can be seen from the table, Yang [12], Sun [25] and our proposed scheme achieve searchability. It is worth noting that these schemes have the characteristics of privacy protection and access control, which is a key security goal of the electronic record sharing system. In addition, regardless whether it is based on the blockchain, only in our scheme, the data is stored in the IPFS, which effectively solves the problem that the data lost or tampered in the cloud environment. And it is fortunate that only our scheme meets all the properties, and our scheme more suitable for current computing systems and practical applications.

6.2 Computational complexity analysis

We present some major time-consuming operational symbols, exponential operations $E$, and bilinear pairwise operations $P$, before analyzing the computational complexity. Since our scheme is based on ABE construction, we use $n_a$ to represent all the number of attributes in the scheme, and $n_{a,u}$ to represent the number of attributes owned by a data requester in the scheme. As it can be seen from Table 3, in the global setup phase, our computational overhead requires only one pair of operations and three exponential operations, and is a constant. In the EMRs encryption phase, since the corresponding access policy is set, the required computing operations are related to the number of attributes. During the user key generation phase, the doctor distributes the attribute private key to the data requester, and the required operations are also related to the number of attributes. In the decryption phase, the data requester needs to calculate two pairs of operations related to the number of attributes.

6.3 Time cost of cryptographic algorithm

In order to analyze the actual time overhead of the cryptosystem, a series of simulation experiments were carried out using a actual data set and a PBC (Pairing-Based Cryptography) library [35]. Because the time cost of some cryptographic algorithms varies with the number of attributes, we simulate some time costs for different number of attributes. In order to better meet the current actual needs, we assume that the number of attributes is $|n_a| = |n_{a,u}| \in [5,25]$, and the experimental results are shown in Table 4 and Fig 6.

We separately analyzed the time cost of some encryption algorithms in the article when the number of attributes is 5, 10, 15, 20, 25. As it can be seen from Table 4 and Fig 6, in the establishment phase of the cryptographic algorithm, with the number of attributes changing, the required time cost is always keep a constant of 21.71 ms, since there is no attribute operation involved in this stage. In the medical record encryption algorithm, when the number of attributes is 5, 10, 15,
20, 25, the time required by the algorithm is 29.238 ms, 38.648 ms, 48.058 ms, 57.468 ms, 66.878 ms, respectively. Compared with the medical record encryption algorithm, the user request algorithm reduces the required time by about 16.064 ms under the corresponding number of attributes. As it can be seen from Fig 6, in the decryption algorithm, since our scheme involves two pairs of operations, the time cost required is slightly longer than other algorithms.

### 6.4 The cost of smart contract

In order to test the gas cost of the smart contract, we implement it on intel(R) Core(TM) i5-2450M CPU2.50GHz and 4.00GB of RAM with Ethereum in solidity [36] code, a programming language for writing contracts on Ethereum. We tested the cost of deploy contract and add requester. In the experiment, we set `gasprice = 1Gwei`. The cost of gas used in smart contract is shown in Table 5. As can be seen from Table 5, the deploy contract operation is performed during the system establishment phase at a cost of 0.000154455 Ether. The add requester operation is performed during the medical data search and access phase, and the operation is performed after the doctor authenticates the identity of the access requester, and the cost of the operation is 0.000047869 Ether.

By analyzing the test results in Table 5, it is not difficult to find that our electronic medical record sharing contract requires less cost in deployment and invocation, and it is acceptable for users. In addition, although the cost of calling some of the functions in the contract increases as the number of electronic medical records increases, the increase is small. Therefore, our scheme is feasible in practice.

### 6.6 System data throughput analysis

In the scheme, the block is composed of a block header and a block body. The size of the block header is about 80 bytes, and a certain number of transactions make up a block body. By
analyzing the data, it is concluded that the size of a transaction is 256 bytes. We assume that a block contains 20 transaction information, the size of a block is calculated as $256 \times 20 + 80 = 5200 \text{bytes} = 5 \text{Kb}$, on this basis we can measure the throughput of the network.

Suppose an EMRs system with 500 users, where the peak number of transactions generated per second is 50 users sending transaction information. Therefore, the throughput of system data can be calculated as follows:

$$50 \times (5200) = 260000 \text{bytes/s} \approx 253.9 \text{Kb/s}$$

$$50 \times (5200 \times 60) = 1560000 \text{bytes/min} \approx 14.98 \text{Mb/min}$$

$$50 \times (5200 \times 60 \times 60) = 936000000 \text{bytes/h} \approx 892.64 \text{Mb/h}$$

$$50 \times (5200 \times 60 \times 60 \times 24) = 22464000000 \text{bytes/day} \approx 20.92 \text{Gb/day}$$

Table 5. Smart contract cost test (gas price = 2 Gwei).

| Function       | Gas used | Gas cost (ether) |
|----------------|----------|------------------|
| Deploy contract| 772275   | 0.000154455      |
| Add requester  | 23948    | 0.000047869      |

https://doi.org/10.1371/journal.pone.0239946.t005
According to the data calculated above, we can calculate the total amount of data of the system over a period of time, so as to estimate the growth of block chain data. The results are shown in Table 6, where the ordinate and abscissa respectively represent the number of transactions and a fixed period of time. The data in the table is calculated based on the number of peak transactions in the network.

Table 6. Total amount of system data in a fixed time.

| Transaction amount | 1 s   | 1 h   | 1 day | 1 year |
|--------------------|-------|-------|-------|--------|
| 500                | 2.479Mb | 8.717Gb | 209.21Gb | 74.57Tb |
| 5000               | 24.79Mb | 87.17Gb | 2.04Tb  | 744.6Tb |
| 50000              | 247.94Mb | 871.72Gb | 20.43Tb | 72.88Pb |
| 500000             | 2.42Gb  | 8.51Tb  | 204.3Tb | 72.88Pb |

https://doi.org/10.1371/journal.pone.0239946.t006

Based on the data calculated above, we can deduce the total amount of data of the system over a period of time, so as to estimate the growth of the amount of data on the blockchain. The results are shown in Table 6, where the ordinate and abscissa represent the number of transactions and a fixed period of time, respectively. The data in the table is calculated based on the number of peak transactions in the network. Compared with the Bitcoin system, in the Bitcoin system, when the transaction volume is 2000, the amount of data generated in one year can reach the Pb level. However, in our system, when the transaction volume reaches 10,000, the amount of data generated in a year can reach the Pb level. Therefore, from the perspective of the amount of data, our scheme has more advantages.

7. Conclusion

We have proposed a framework using blockchain and smart contract technology to solve the problem of secure storage and sharing of current EMRs. The system enables doctors firstly encrypt electronic medical records with appropriate access policies and then upload the ciphertext to IPFS. The combination of IPFS and blockchain allows doctors to process large amounts of electronic medical data via IPFS, to eliminate the need of putting the data itself on the chain, to save network bandwidth in the blockchain. While using doctors and patient encryption to achieve electronic medical record security, a keyword index for searching encrypted medical records is also designed. The encrypted keyword index data is stored in the Ethereum blockchain, and the smart contract is deployed on the Ethereum blockchain to perform keyword search in the distributed system. And we use ABE to achieve fine-grained access control for data requester. Security analysis shows that the protocol implements data security, privacy protection, and secure search. In addition, we evaluated performance from three aspects: the characteristics of the scheme, the time cost of cryptographic algorithm, and the smart contract cost. In future work, we will address the more complex needs of EMRs system, for instance, multi-keyword search and time-controlled revocation.

Supporting information

S1 Appendix.

(DOC)

Author Contributions

Supervision: Xiaomin Yao.
Writing – original draft: Lili Ren.  
Writing – review & editing: Jin Sun, Shangping Wang.

References
1. Danan T, Shiping C, Surya N; Secure multiparty data sharing in the cloud using hardware-based TPM devices; Jun. 2014, pp. 224–231.
2. Nasir K A, Miss L M, Mazhar A, Saied M: BSS – Block-based sharing scheme for secure data storage services in mobile cloud environment; J. Supercomput., vol. 70, no. 2, pp. 946–976, Nov. 2014.
3. Xin D, Jiadi Y, Yuan L, Ying C, Guan X, Ming L: Achieving an effective, scalable and privacy-preserving data sharing service in cloud computing; Comput. Secur., vol. 42, pp. 151–164, May 2014.
4. Yang J J, Li J Q, and Niu Y: A hybrid solution for privacy preserving medical data sharing in the cloud environment in Future Generat. Comput. Syst., vols. 43–44, pp. 74–86, Feb. 2015.
5. Au M H, Yuen T H, Liu J K. A general framework for secure sharing of personal health records in cloud system[J]. Journal of Computer and System Sciences, 2017: S0022000017300296.
6. Nakamoto S. Bitcoin: A peer-to-peer electronic cash system, 2008.
7. Guy Z, Oz N, Alex S P: Decentralizing privacy: Using blockchain to protect personal data; May 2015, pp. 180–184.
8. Peterson K, Deedvurana R, Karjamala P, Boles K: A blockchain-based approach to health information exchange networks; 2016, pp. 1–10.
9. Krawiec R J. Blockchain: Opportunities for health care; 2016, pp. 1–16.
10. Qi X, Emmanuel BS, Kwame O A, Gao J, Du X, Mohsen G: MeDSHare: Trust-less Medical Data Sharing Among Cloud Service Providers Via Blockchain; vol. 5, July 2017, pp. 14757–14767.
11. Zhang A, Lin X: Towards Secure and Privacy-Preserving Data Sharing in e-Health Systems via Consortium Blockchain; vol. 42, no. 8, 2018:140.
12. Yang Y, Ma M: Conjunctive Keyword Search with Designated Tester and Timing Enabled Proxy Re-encryption Function for E-health Clouds[J]: IEEE Transactions on Information Forensics and Security; 2016, pp.746–759.
13. Blockchain for Financial Services. Accessed: Mar. 25, 2018. [Online]. Available: https://www.ibm.com/blockchain/financial-services
14. Blockchain for Supply Chain. Accessed: Mar. 25, 2018. [Online]. Available: https://www.ibm.com/blockchain/supply-chain
15. A Decentralized Network for Internet of Things. Accessed: Mar. 25, 2018. [Online]. Available: https://iotex.io
16. Fernández-Caramés T M; Fraga-Lamas P: A review on the use of blockchain for the internet of things; IEEE Access, 2018.
17. Nick S; The idea of smart contracts; Nick Szabo’s ’Zs Papers and Concise Tutorials, vol. 6, 1997.
18. Ibraimi L, Asim M, Petkovic M: Secure Management of Personal Health Records by Applying Attribute-Based Encryption; IEEE, 2009.
19. Li M, Yu S, Zheng Y: Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24 (1):131–143.
20. Fatos X, Li J, Zhao G, Chen X, Wong D S: Designing cloud-based electronic health record system with attribute-based encryption; vol. 74, no. 10, 2015, pp. 3441–3458.
21. Kim H, Song H, Lee S: A simple approach to share users’ own healthcare data with a mobile phone. Eighth International Conference on Ubiquitous and Future Networks (ICUFN). IEEE, 2016.
22. Qi X, SIFAHE B, Amofa A; BBDS: Blockchain-Based data sharing for electronic medical records in cloud environments; vol. 8, no. 44, 2017, pp. 1–16.
23. Ramani V, Kumar T, Bracken A, Liyanage M, Ylianttila M: Secure and efficient data accessibility in blockchain based healthcare systems. GLOBECOM, Dec. 2018. pp. 206–212.
24. Sun J, Wang X, Wang S, Ren L (2018) A searchable personal health records framework with fine-grained access control in cloud-log computing; PLoS ONE 13(11): e0207543. https://doi.org/10.1371/journal.pone.0207543 PMID: 30496194
25. Zhang P, White J, Schmidt D C, Gunther L, Rosenbloom S T; FHIRChains: Applying Blockchain to Securely and Scalably Share Clinical Data. vol. 16, 2018, pp. 267–278.
26. Yaeger K; Emerging Blockchain Technology Solutions for Modern Healthcare Infrastructure. vol. 2, no. 1, 2019, https://doi.org/10.29024/jsim.7

27. Nguyen D C, Pathirana P N, Ding M, Seneviratne A. Blockchain for Secure EHRs Sharing of Mobile Cloud Based E-Health Systems. IEEE Access. 2019. pp. 66792–66806.

28. Juan B; IPFS-content addressed, versioned, P2P file system. [Online]. 2014, Available: https://arxiv.org/abs/1407.3561

29. Amit S, Brent W; Fuzzy identity-based encryption. Springer, 2005, pp. 457–473.

30. Adi S; Identity-based cryptosystems and signature schemes. Springer, 1985, pp. 47–53.

31. Goyal V, Pandey O, Amit S, Brent W; Attribute-based encryption for fine-grained access control of encrypted data. New York, NY, USA: ACM, 2006, pp. 89–98.

32. Li H, Zhang F, He J, Tian H; A Searchable Symmetric Encryption Scheme using BlockChain, 2017.

33. Miao Y, Ma J, Jiang Q, Li X; Verifiable keyword search over encrypted cloud data in smart city. Computers & Electrical Engineering, vol. 65, 2017, pp. 1–12.

34. Jacob T S; Fast probabilistic algorithms for verification of polynomial identities in Journal of the ACM (JACM), vol. 27, no. 4, 1980, pp. 701–717.

35. Duquesne S, Lange T (2005) Pairing-based cryptography. Mathiscernet volume 22: 573–590.

36. Solidity. Accessed: Jun. 10, 2017. [Online]. Available: http://solidity.readthedocs.io/en/latest