Article | Received 4 March 2025; Accepted 27 April 2025; Published 27 May 2025
https://doi.org/10.55092/blockchain20250012

A blockchain-based access control method for large-scale electronic medical records

Zhen Chu[1], Wangjie Qiu[1,2,*], Tianyu Lei[2], Jingchun He[2] and Qinnan Zhang[2]

[1] Institution of Medical Artificial Intelligence, Binzhou Medical University, Yantai 264003, China

[2] Institute of Artificial Intelligence, Beihang University, Beijing 100191, China

* Correspondence author; E-mail: wangjieqiu@buaa.edu.cn.

Highlights:

- A flexible access control solution.
- Automation in access control.
- A detailed logic and workflow for EMR interaction and sharing.

Abstract: The widespread adoption of emerging technologies in healthcare has led to an exponential increase in medical data generation. However, the security of healthcare data has not kept pace, with frequent breaches and unauthorized access posing substantial threats to patient privacy and the integrity of healthcare systems. Although existing access control frameworks offer partial solutions for secure data access, they fall short in authorization granularity, privacy preservation, and large-scale, high-frequency access. To bridge these critical gaps, we propose a novel role-based access control (RBAC) framework that enables secure and efficient management of large-scale, high-frequency data access. The framework first introduces a real-time access behavior analysis algorithm. It then integrates Ethereum smart contract technology with the RBAC model to construct high-performance, scalable access control contracts. Subsequently, the framework simulates the EMR interaction process in a representative healthcare scenario. Through rigorous security evaluations and experimental simulations, we demonstrate that the proposed framework enables robust accessor management, secure data sharing, and effective support for large-scale, high-frequency access while maintaining operational efficiency. This work offers a scalable and practical solution to healthcare data security in the era of big data and population aging.

Keywords: blockchain; electronic medical records; smart contracts; access control

1. Introduction

The integration of technologies such as the Internet of Things (IoT), medical wearables, virtual reality, and large-scale models into the healthcare sector is progressively facilitating the development of

more human-centered "smart healthcare" services [1]. Simultaneously, intelligent medical devices generate more detailed EHR data, enabling continuous monitoring of patient health and personalized treatment recommendations. In the era of artificial intelligence, sharing medical data can facilitate multi-organizational research, enhance disease prevention efforts, and contribute to the development of standardized regulations and guidelines for the healthcare industry. However, unlike other types of data, medical data is often highly sensitive, raising concerns about privacy breaches. This sensitivity has led to medical data being isolated in "data silos," reducing the willingness to share [2,3]. In numerous data leakage incidents, the primary causes include improper access control settings, unauthorized user access, and fraudulent behavior by third-party entities, with some cases also attributable to system framework vulnerabilities exploited by hackers [4].

From the perspective of traditional access control models—such as Role-Based Access Control (RBAC) [5], Attribute-Based Access Control (ABAC) [6], Capability-Based Access Control (CapBAC) [7], and Access Control Lists (ACL) [8]—applied to healthcare scenarios, a disheartening reality emerges: these models are predominantly centralized, necessitating trusted institutions to act as intermediaries. With the advancement and widespread adoption of blockchain technology, we now have access to a decentralized, distributed, and auditable foundational framework [9]. Its immutability, fairness, and transparency have garnered significant acclaim, positioning the integration of blockchain and access control technologies in healthcare as a rapidly growing research area. However, this integration introduces new challenges: the fairness of post-verification execution cannot be fully assured, and severe fraudulent behavior may significantly undermine the system's control effectiveness. In 2014, Ethereum's smart contracts gained widespread attention due to their automated execution capabilities, particularly in the finance and insurance sectors [10]. These self-executing programs require no human intervention, ensuring transparency and safeguarding the interests of all parties. Recent research on smart contract-based access control models has revealed that the healthcare sector, due to its large data volumes, frequent access requests, diverse user base, and stringent data security requirements, still lacks a scalable, secure, efficient, and patient-centric access control model. In fact, an effective access control model must not only prevent data leakage but also mitigate the widespread adverse impacts in the event of a security breach.

This study addresses the challenge of securing medical data while enabling large-scale and efficient access via a smart contract-based access control system, thereby facilitating secure and efficient sharing between data owners and users. First, we developed an algorithm to analyze access behaviors associated with electronic health records. Concurrently, to enhance user management efficiency, we designed specific roles based on access requirements. Subsequently, we integrated Ethereum blockchain technology with the RBAC model to construct a novel smart contract tailored to these requirements. Finally, we employed proxy re-encryption technology to demonstrate EHR storage and sharing interactions in general scenarios, thereby highlighting the flexibility of our model. In applying smart contract-based access control to the healthcare domain, this study introduces an innovative user categorization scheme based on access roles, addressing both control direction and data classification by application value. This approach enables efficient interaction between user and data, while also ensuring data security and sensitivity during sharing. The goal of this study is to establish secure and trustworthy network communication among all stakeholders, leveraging the proposed access control model to support the business logic of data sharing

in healthcare scenarios.

The structure of the paper is as follows: Section 2 reviews and analyzes related work. Section 3 presents the necessary background knowledge. Section 4 describes the design and implementation of the proposed smart contract-based access control framework for healthcare applications. Section 5 presents the experimental results along with a security analysis involving all participating entities. Finally, Section 6 concludes the paper.

2. Related work

This section reviews research methodologies concerning blockchain-based access control, particularly those applied to electronic health records. Access control is a fundamental component of information systems, responsible for ensuring security by verifying user authorization for requested services. Existing access control schemes face persistent challenges in security, privacy protection, and reliance on trusted third parties. Blockchain technology offers a promising solution by enabling decentralized enforcement and management of access permissions through smart contracts.

2.1. Blockchain-based access control schemes

Zyskind [11] proposed a decentralized data management model centered on data owners, transforming the blockchain into a trusted third-party access controller. However, this model has limitations in secure data transmission, as the encryption process is stored in a trusted third party, which cannot guarantee data security. Wang [12] combined attribute-based encryption technology with the InterPlanetary File System (IPFS) to propose a blockchain-based data storage model supporting fine-grained access. Nevertheless, this scheme cannot handle the revocation of old user attributes or the addition of new attributes, and the access policy is immutable. Xu integrated a capability-based access control model with blockchain, proposing a model that stores data on the blockchain. However, this model does not consider the overhead and system latency of blockchain, nor is it suitable for storing large and diverse file formats. Ouaddah [13] proposed FairAccess, a blockchain-based access control model for the Internet of Things (IoT), which manages data in a user-centric manner to achieve decentralized access control. However, this model creates unique access policies for each user, leading to policy redundancy and management difficulties. Wang designed an attribute-based distributed access control model for lightweight IoT devices, storing access control policies, objects, and subject attributes in smart contracts to implement control logic. However, this model does not verify the pressure relationship between large-scale IoT devices and access control. Putra [14] proposed a blockchain and smart contract-based IoT access control model, combining a main blockchain with multiple sidechains, providing a new approach for large-scale complex systems. Ullah [15] proposes a blockchain-based audit framework for electronic health record (EHR) access, integrating smart contracts with the Purpose-Based Access Control (PBAC) strategy to address the lack of effective auditing in medical data security. The system achieves transparent logging and real-time monitoring of EHR access behaviors by constructing an immutable audit trail, enforcing compliance auditing of access control policies, and generating structured audit logs.

Rashid [16] proposed ACS-IoT, an enterprise-level access control system for the Internet of Things (IoT) based on smart contracts and blockchain. In this system, all IoT devices are registered on the

blockchain network and managed without requiring a centralized authority or validation by a central management center. This approach achieves fine-grained access control over internal enterprise data. Tian [17] proposed MSLShard, a lightweight and secure framework that employs an adaptive network sharding scheme based on network distance, node credibility, and access frequency. This design alleviates node storage pressure and enhances scalability. MSLShard achieves improved scalability and security in large-scale IoT environments by incorporating blockchain storage from a sharding perspective. Wang [18] proposed Hierarchical Blockchain Enabled Cloud-Edge Collaborative Federated Learning (HBCE-FL), a framework that utilizes multi-level access control encryption to share IoT knowledge. User privacy requirements are classified into multiple levels, enabling fine-grained privacy protection. Experiments conducted on two datasets demonstrate that the framework supports federated multi-task learning with personalized privacy guarantees.

2.2. Blockchain-based electronic health record (EHR) access control schemes

Tang [19] proposed a blockchain-based EHR authentication scheme using identity-based signatures. However, this scheme does not consider the security of medical data generated by smart medical devices. The MedChain model utilizes blockchain to store and share medical data, relying on its immutability to ensure data security. Patients, doctors, and healthcare providers share data with each other through blockchain technology and peer-to-peer networks. These methods collaborate to provide a flexible and efficient data-sharing system. Nevertheless, this model requires uploading medical data to the blockchain, leading to increased data transmission overhead and verification delays. Jabbar [20] designed the BliMed model to address EHR sharing issues. This model proposes a data storage and blockchain network framework, where the data storage system is responsible for data collection, storage, and transmission, while the blockchain network maintains network operations, uses smart contracts for access control decisions, and ensures data integrity through hash algorithms. However, this scheme still adopts traditional data ownership, and patients do not have actual control over their data. Saini [21] employed smart contracts on a private Ethereum network to manage EHRs. EHRs contain sensitive personal data of patients, so security and privacy issues must be considered when handling such data. To this end, an EHR sharing model was designed to facilitate the sharing of medical data among patients, healthcare institutions, and other entities. This model encrypts EHRs and uses smart contracts to implement access control functions, ensuring the privacy of EHRs while utilizing cloud storage to store encrypted EHR data. The proposed scheme is patient-centric, enabling real-time monitoring of EHRs. However, the design of access control policies using smart contracts in this scheme is redundant. Abid [22] put forward an access control framework for an Internet of Things (IoT)-based intelligent medical system based on smart contracts. This framework utilizes smart contracts to offer distributed and reliable access control, taking into consideration time authorization constraints. Meanwhile, a security attribute analysis of the smart contracts was carried out, and no vulnerabilities were identified. The cost of access control operations can yield favorable outcomes with relatively low Gas costs and delays. Nevertheless, the cost assessment related to multi-linear contract invocations was not taken into account. Psarra [23] proposed a context-aware and blockchain-based access control mechanism that integrates application prediction models to access sensitive electronic health records. Contextual information is leveraged to

detect critical situations, while neural networks (NNs) are employed to forecast patients' health indicator values for the following two hours. This mechanism enables emergency clinicians to securely access sensitive data, supporting timely medical interventions while safeguarding patient health. Practical validation demonstrates that the access control mechanism can be effectively deployed in real-world hospital environments. The system integrates fuzzy logic and predictive machine learning techniques within private and permissioned blockchain networks to proactively alert emergency teams—composed of professional clinicians—about patient emergencies. Additionally, it leverages the decentralized nature of blockchain to ensure data integrity and security, thereby enhancing user trust in the access control mechanism. Ullah [24] proposed a security management framework for electronic health records (EHRs) that integrates Purpose-Based Access Control (PBAC) with blockchain smart contracts, and utilizes run-length encoding (RLE) for the lossless compression of COVID-19 lung CT images, aiming to reduce storage overhead. Experimental results demonstrate that the proposed scheme achieves a compression rate of 45%–68% while preserving diagnostic image quality. Furthermore, it enables transparent data access control and automated payment management via smart contracts. This research offers a practical solution for the efficient storage and secure sharing of medical data. However, the effectiveness of RLE may be limited in scenarios requiring high-fidelity image preservation.

2.3. Performance comparison

The table1 presents the results of the seven indicators for the electronic medical record access control framework. This paper compares this scheme with other existing models in the table. The comparison concludes that this scheme offers distinct advantages over others, particularly in its modifiable access strategy for patient ownership and its automated decision-making through smart contracts. Furthermore, it meets the demands for large-scale, frequent, and efficient access.

Table 1. Index comparison.

Metrics	**ref[12]**	**ref[13]**	**ref[21]**	**ref[22]**	**Ours**
Decentralization	✓	✓	✓	✓	✓
Authentication	✓	✓	✓	✓	✓
Privacy Protection	✓	✓	✓	✓	✓
Third-party Storage	✓	×	✓	✓	✓
Modifiable Access Policy	×	×	×	✓	✓
Based on Smart Contracts	×	×	✓	×	✓
Patient Ownership	×	×	×	×	✓

3. Preliminaries

3.1. Blockchain

Since the introduction of Bitcoin—a revolutionary cryptocurrency—blockchain technology has rapidly evolved into a versatile, interdisciplinary innovation [9]. Fundamentally, blockchain is a distributed ledger comprising a sequence of interconnected blocks. Each block contains metadata and transaction records, and the use of cryptographic hash algorithms ensures immutability—each block maintains

integrity and consistency by including the hash of the preceding block. Blockchain exhibits five fundamental characteristics:

(1) Decentralization: Traditional models rely on a centralized trusted authority for transaction verification, making them susceptible to single points of failure and increasing maintenance overhead. In contrast, distributed systems such as blockchain—despite inherent data redundancy—eliminate dependence on a centralized authority, thereby reducing operational costs and mitigating system performance bottlenecks.

(2) Immutability: Blockchain systems employ cryptographic hash algorithms to guarantee that once transaction data is recorded, it becomes tamper-proof. This immutability is fundamental to ensuring robust auditability within the system.

(3) Transparency: In blockchain systems, all authorized users can access and verify historical transaction records. This inherent transparency greatly facilitates auditing processes.

(4) Traceability: Each block is embedded with a timestamp indicating its creation time, and every transaction is similarly time-stamped. This time-stamping mechanism enables users to trace the chronological sequence of transactions.

(5) Non-repudiation: Every transaction must be signed using the initiator's private key at the time of initiation. The corresponding public key allows other users to verify the signature's authenticity, thereby ensuring the transaction's validity.

3.2. Smart contracts

The concept of smart contracts was first introduced by Nick Szabo [25] in 1997, with the goal of automating the execution of legal agreements. Today, smart contracts have evolved into automated scripts that are executed synchronously across multiple nodes in distributed systems, such as blockchains. This technology eliminates the need for external trusted institutions, enabling mutually distrustful nodes to execute contract code with precision. Users can deploy smart contracts on the Ethereum network, define their business logic, and have it automatically invoked and executed by the Ethereum Virtual Machine (EVM). Essentially, smart contracts are programs built on blockchains, originating from verified contractual terms. Smart contracts implement appropriate access control mechanisms and ensure the execution of agreements, including the allocation of access permissions for each function within the contract. The execution sequence of a contract is deterministic, ensuring predictable outcomes. Once the conditions specified in the smart contract are met, the triggered statements will automatically execute the corresponding functions in a predictable manner.

3.3. Proxy re-encryption

Proxy re-encryption [26] is a cryptographic scheme that utilizes a trusted third party or semi-honest intermediary to convert ciphertext, encrypted with the data owner's public key, into a format that can be decrypted by the data requester's private key. This technology enables the secure sharing of encrypted data while ensuring the protection of user privacy. The specific process is as follows:

(1) Key generation:

The data provider generates data M and uses the KeyGen function to create local public-private key pairs for both the data provider and the data requester:

- Data provider's key pair:

$$(pk_A, sk_A), \quad pk_A = g^{sk_A} \tag{1}$$

- Data requester's key pair:

$$(pk_B, sk_B), \quad pk_B = g^{sk_B} \tag{2}$$

(2) Data encryption:

The data provider uses the Encrypt function to encrypt the plaintext M and uploads the ciphertext C to the proxy re-encryption server (proxy):

- Select a random number r.
- Compute the ciphertext:

$$C = (C_1, C_2) = (g^r, M \cdot pk_A^r) \tag{3}$$

- Send (pk_A, C) to proxy.

(3) Re-encryption key generation:

The data requester provides their public key pk_B to the data provider. The data provider then uses the ReGenKey function to generate a re-encryption key rk for the data requester and sends it to proxy:

- Obtain the data requester's public key pk_B.
- Generate the re-encryption key:

$$rk = \frac{sk_A}{H(pk_B^{sk_A})} \tag{4}$$

- Send rk to proxy.

(4) Re-encryption:

Proxy uses the ReEncrypt function, along with the ciphertext C and the re-encryption key rk, to generate a new ciphertext C':

- Receive (pk_A, C, rk) from the data provider.
- Compute the re-encrypted ciphertext:

$$C' = (C_1', C_2') = (C_1^{rk}, C_2) \tag{5}$$

(5) Decryption and verification:

The data requester obtains the re-encrypted ciphertext C' and decrypts it using their private key sk_B:

- Receive the ciphertext C'.

- Compute the plaintext:

$$M' = \frac{C_2'}{(C_1')^{H(pk_A^{sk_B})}} \tag{6}$$

- Verify the decryption:

$$M' = \frac{C_2'}{(C_1')^{H(pk_A^{sk_B})}} = \frac{C_2}{C_1^{rk \cdot H(pk_A^{sk_B})}} = \frac{C_2}{C_1^{sk_A}} = \frac{M \cdot pk_A^r}{g^{r \cdot sk_A}} = \frac{M \cdot g^{sk_A \cdot r}}{g^{r \cdot sk_A}} = M \tag{7}$$

4. Access control model

4.1. Model overview

This section presents a novel blockchain-based access control system for electronic health records, designed to support data preservation, large-scale access, and cross-organizational data sharing. First, we provide an overview of the system architecture and define its security objectives. Second, we describe the functional roles of the nodes participating in the system. Then, we introduce a novel access behavior analysis algorithm and integrate it into smart contracts to enforce access control. Finally, we describe the access workflow, demonstrating the interaction process among the nodes.

The system is built on Ethereum's peer-to-peer network and smart contract infrastructure. Each node enforces access control through smart contracts, with policies implemented in Solidity—a Turing-complete programming language—to validate access requests. Industry research revealed that while the number of access requesters is large and access attempts are frequent, the types of data and user structures are relatively homogeneous. To address this, we redesigned an access behavior analysis algorithm grounded in the traditional Role-Based Access Control (RBAC) model. The algorithm adopts a two-layer architecture, enabling streamlined yet efficient access policies that meet the general data-sharing requirements of the healthcare sector. However, we observed that overly simplified and efficiency-driven policies may compromise patient-centric principles in access control. To resolve this issue while maintaining rapid and authorized access, a patient-centric interaction model was adopted. In summary, the proposed model integrates smart contract technology, proxy re-encryption, and the RBAC mechanism. This model is expected to serve as a cross-domain access control solution, effectively addressing scenarios characterized by large data volumes, high-frequency access, and homogeneous data types and user structures.

To ensure the security of electronic health records during sharing and storage, this proposal outlines the following security objectives:

(1) Security of EHR Sharing: Throughout the EHR sharing process, data remains encrypted during transmission, thereby ensuring that only authorized users can access it.

(2) Legitimacy of Regulatory Verification and New User Addition: The process through which regulators verify institutions and register new accessors must comply with legal standards, thereby preventing fraudulent behaviors such as deception.

(3) Traceability and Regulatory Capability in EHR Sharing: The EHR sharing process must ensure traceability and facilitate regulatory oversight across the entire data-sharing lifecycle, thereby enabling effective supervision in the event of disputes or malicious behavior.

4.2. Model node composition

The network comprises six primary entities—Governing Organization (GO), Institution (IN), Agent (AG), Requester (RE), Patient (PA), and Storage Node (ST)—as illustrated in Figure1. The roles and responsibilities of each entity are detailed below:

(1) Governing Organization (GO): Within the healthcare context, the GO is tasked with overseeing the entire medical ecosystem. Its core responsibilities include institutional management and supervision of the data-sharing process. Designated administrators are authorized to categorize institutions, verify their legitimacy, assign role-based attributes, maintain a list of trusted entities, add new institutions, and deploy access control policies.

(2) Patient (PA): Patients obtain medical services from institutional nodes equipped with healthcare functionalities, resulting in the generation of EMR data. They retain full control over their EMR files.

(3) Institution (IN): Institutions are entities registered and authorized by the GO, having attained a recognized level of trust. Typical examples include healthcare providers, insurance firms, and pharmaceutical companies. They are responsible for directly managing their associated requesters and registering them within smart contracts.

(4) Agent (AG): During the data storage phase, the AG transfers encrypted data provided by authorized healthcare institutions to the designated storage provider and maintains a mapping of file storage addresses to their corresponding ownership. During the access phase, the AG receives proxy re-encryption keys from patients, performs re-encryption on the original ciphertext using these keys, and delivers the resulting ciphertext to the requester.

(5) Requester (RE): Requesters are associated with a specific institution and represent the demand side for electronic medical records, with the ability to initiate requests for accessing EMR data.

(6) Storage Node (ST): The ST represents a persistent data storage system, such as IPFS or cloud storage, which is capable of storing various file types and primarily interacts with the AG.

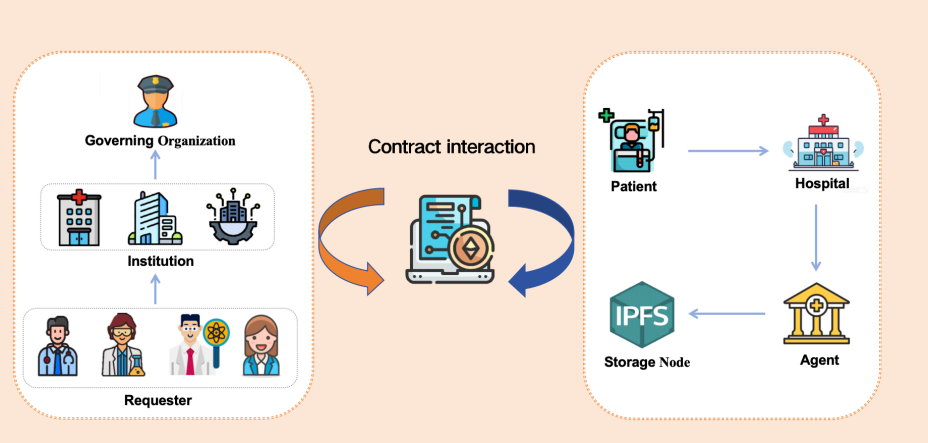


Figure 1. Electronic medical record sharing system.

4.3. Access behavior analytical algorithm

The access behavior analysis algorithm is built on a two-tier analytical framework, consisting of basic behavior analysis and permission matching analysis. The basic behavior analysis assesses the roles

of requesters in relation to the access permissions of electronic medical records, while the permission matching analysis performs a secondary evaluation to verify whether the permission subset of a role corresponds to the permissions of the EMRs.

Based on a review of existing electronic medical records, we have categorized them into distinct classes, as shown in Table 2. The roles and significance of these categories are discussed in detail below.

Table 2. Data hierarchy and role-permission mapping.

Data Level	Data Type	Data	Role-Permission Mapping
S1	Record Type	Medical records, treatment records, admission and discharge records	Healthcare institutions
	Data Type	Laboratory test reports, imaging test reports	Healthcare institutions, Research institutions
	Medication Type	Prescription drugs, medical orders	Healthcare institutions, Pharmaceutical companies, Research institutions
S2	Insurance Type	Payment receipts, insurance certificates	Healthcare institutions, Medical insurance
	Summary Type	Case summaries	Public health

Record type: Files in this category primarily document patients' basic information, medical history, and treatment processes, assisting physicians in formulating personalized treatment plans. Additionally, these detailed records serve as evidence in the event of medical disputes. Key documents include: medical records (which document patient history, diagnoses, and treatment processes), treatment records (which include prescriptions and nursing notes to facilitate subsequent adjustments to treatment plans), and admission and discharge summaries (which summarize the patient's condition upon admission and discharge).

Data type: Files in this category are generated by specialized medical instruments, providing definitive test results and data that support evidence-based treatment decisions and further medical research and disease studies. Key documents include test reports, such as laboratory test reports and imaging examination reports.

Medication type: Files in this category document information about medications administered during treatment, provide relevant treatment details during patient referrals, and offer data to support pharmaceutical research and development. Key documents include medication records, which track drug usage and dosage throughout the treatment process.

Insurance type: Files in this category are used for medical insurance reimbursement and payment verification, thus requiring their classification as a distinct category. Key documents include invoices, payment receipts, and other financial records.

Summary type: Files in this category provide a comprehensive overview of the patient's entire treatment process, summarizing medical records to help physicians understand the patient's overall health background. They also ensure that medical research adheres to ethical standards and protects patient privacy. Key documents include treatment summaries and brief medical histories.

The insurance and summary categories of EMRs are stored at the first hierarchical level, while the

record, data, and medication categories are stored at the second hierarchical level. The first level employs the symmetric encryption algorithm AES-128 for secure storage, whereas the second level utilizes the asymmetric encryption algorithm AES-256 for enhanced security. For the hierarchically encrypted EMR data, storage is implemented layer by layer. Let the EMR dataset be defined as $D = \{D_1, D_2, \ldots, D_n\}$, where D_i represents the i-th EMR data. For each EMR data D_i, encryption is applied based on the privacy level of each data item, as follows:

$$D_i = \{D_j^{rout}, D_j^{susc}\},$$

where D_j^{rout} denotes the routine medical information in the i-th EMR data, encrypted using the symmetric AES-128 algorithm, and D_j^{susc} represents the sensitive medical information in the i-th EMR data, encrypted using the asymmetric ECC algorithm. The encrypted data is then stored hierarchically as follows:

$$S_1 = \{D_1^{rout}, D_2^{rout}, \ldots, D_k^{rout}\},$$

where S_1 represents the data storage layer for routine medical information,

$$S_2 = \{D_1^{susc}, D_2^{susc}, \ldots, D_m^{susc}\},$$

where S_2 represents the data storage layer for sensitive medical information. For routine medical information, if the data is accessed frequently during the current patient's treatment period, the encryption strength is increased, and the data is elevated from the routine storage layer (S_1) to the sensitive storage layer (S_2). Conversely, for sensitive medical information, if the data remains unaccessed during the current patient's treatment period, the encryption strength is reduced, and the data is downgraded from the sensitive storage layer (S_2) to the routine storage layer (S_1).

In the system node composition, we classify one type of node as "institutions," which manage numerous requesters. Accordingly, roles are assigned based on the specific access needs of these institutions. Below, we provide a detailed description of the roles assigned to institutions and their associated functions.

Healthcare institutions: These include hospitals, medical examination centers, and other facilities that cater to patients' healthcare needs. Typically, medical institutions play a primary role throughout the treatment process. Therefore, we grant this role full access to all file attributes.

Public health: These departments use data for disease surveillance, epidemiological research, and public health policy formulation. They monitor large-scale disease outbreaks in real time and provide early warnings. Therefore, we allocate only summary-level attributes to this role.

Research institutions: Researchers use medical data to study disease mechanisms, develop new drugs, and conduct clinical trials. The value of medical data is crucial to the development processes of researchers, as it is closely linked to the advancement of future healthcare standards. Therefore, we assign data-level attributes to this role.

Insurance companies: Insurance companies analyze medical data to assess insurance risks, design health insurance products, and review claims. Therefore, we allocate insurance-related attributes to this role.

Pharmaceutical companies: Pharmaceutical enterprises use medical data for market analysis, drug development, and marketing strategy formulation. They play a critical role throughout the entire drug development lifecycle. Therefore, we assign drug-related attributes to this role.

Define the set of access roles R as:

$$R = \{r_1, r_2, \ldots, r_n\},$$

where R represents the constructed set of roles, and r_1 denotes distinct roles in the set, including medical institutions, public health departments, research institutions, insurance companies, and pharmaceutical companies.

Additionally, define the set of access permissions for electronic medical records (EMRs) corresponding to each role as:

$$P = \{P_1, P_2, \ldots, P_n\},$$

where P represents the set of EMR access permissions assigned to each role, including permissions for record, data, medication, insurance, and summary categories. Here, P_i denotes the set of EMR access permissions corresponding to role r_i.

Based on the defined sets of roles and permissions, the relationship between roles and permissions is established through a mapping function:

$$M : R \to 2^P,$$

where M represents the mapping between roles and permissions. For each role $r \in R$, there exists a corresponding subset of permissions $P_r \subseteq P$, where P_r denotes the set of permissions assigned to role r. The specific mappings are as follows:

$$\begin{gathered}
M(\text{Medical Institutions}) = \{\text{Record Class}, \text{Data Class}, \text{Drug Class}, \\
\text{Insurance Class}, \text{Summary Class}\}, \\
M(\text{Research Institutions}) = \{\text{Data Class}, \text{Drug Class}\}, \\
M(\text{Pharmaceutical Institutions}) = \{\text{Data Class}, \text{Drug Class}\}, \\
M(\text{Insurance Companies}) = \{\text{Insurance Class}\}, \\
M(\text{Public Health Departments}) = \{\text{Summary Class}\}.
\end{gathered}$$

4.4. Smart contract

The smart contract system comprises five key components: the Access Smart Contract (ASC), the Control Smart Contract (CSC), the Supervision Smart Contract (SSC), the Institution Smart Contract (ISC), and the Policy Smart Contract (PSC). As illustrated in Figure 2, the interaction process among these contracts begins when the ASC receives a target request from an accessor. Upon receiving the request, the ASC invokes the CSC to evaluate the access. The CSC then retrieves institutional legitimacy from the SSC, verifies the accessor's legitimacy through the ISC, and obtains access behavior analysis and decision-making criteria from the PSC. Based on this comprehensive evaluation, the CSC formulates an access decision and returns it to the ASC.

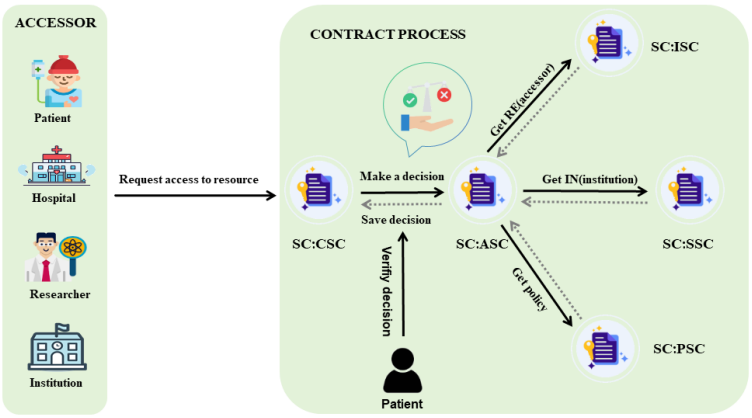


Figure 2. Smart contract interaction process.

The implementation of access control in smart contracts primarily includes the following components: ASC (Access Execution Contract) for executing access processes, CSC (Access Decision Contract) for making access decisions, SSC (Authorization Management Contract) for managing authorized institutions and their attributes, ISC (Visitor Management Contract) for managing visitors affiliated with institutions, and FSC (Access Behavior Analysis Contract) for storing strategies related to access behavior analysis. Table 3 outlines the input parameters of these smart contracts, while Table 4 describes the names of the smart contracts, their associated functions, and the corresponding conditions. Within each function of the smart contracts, a condition is embedded to verify whether the node is permitted to execute the function. If the condition evaluates to true, the function is automatically executed; otherwise, the call fails. These five smart contracts encompass both function code and corresponding databases, which are utilized to store and manage execution records. Functions can save, modify, add, and delete variables within the database. If a function execution fails, no changes are made to the system. In the following sections, we will provide a detailed description of the specific functionalities of these smart contracts.

(1) Access Smart Contract (ASC): The ASC smart contract comprises four functions:

ASC:RequestAccess handles data access requests.

ASC:SaveDecision stores the processing results.

ASC:RevokeAccess revokes the current access request.

ASC:VerifyDecision verifies the legitimacy of related requests.

When RE invokes ASC:RequestAccess, this function triggers a call to CSC:EvaluateRequest. After evaluating the access request, the system makes a decision and subsequently calls back ASC:SaveDecision. In cases where RE exhibits risky behavior, ASC:RevokeAccess can be invoked by the patient, IN, or GO to revoke the RE file permissions and add them to a blocklist. ASC:VerifyDecision is called by the patient to verify whether RE holds valid access rights.

Table 3. Smart contract parameter representation.

Term	Description
$File_{ID}$	the unique identifier of the electronic medical record file
RE_{addr}	requester's address
IN_{addr}	institution's address
P_n	file attributes
$Decision$	whether the access request is allowed or denied
R_n	role of the institution
$RE_{message}$	contains a set of requester's information
GO_{sig}	administrator signs the target information with their private key
IN_{sig}	institution signs the target information with their private key
$Policy_{id}$	identifier of the decision policy
M	the mapping between roles and permission

(2) Control Smart Contract (CSC): The CSC is responsible for executing the decision-making functions for access requests. This contract contains only one function: CSC:EvaluateRequest.

This function is called by ASC:RequestAccess and, during the evaluation process, invokes functions from SSC:AcquisitionInstitution, ISC:AcquisitionUser, and FSC:AcquisitionFilePolicy to retrieve necessary information. Upon completion of the verification, it calls back ASC:SaveDecision.

(3) Supervision Smart Contract (SSC): The SSC manages institutions authorized by regulatory bodies. This contract includes the following functions:

SSC:AddInstitution and SSC:RemoveInstitution for adding and removing institutions.

SSC:RetrievePK and SSC:IsAddrOfInstitution for retrieving the institution's address and verifying the legitimacy of the supervisor's signature.

SSC:AddInstitutionAttr and SSC:GetInstitutionAttr for authorizing institutional attributes and providing access to these attributes.

SSC:AcquisitionInstitution is called by CSC during requests to obtain evaluation criteria.

(4) Institution Smart Contract (ISC): The ISC manages users under its jurisdiction. This contract includes the following functions:

ISC:AddUser and ISC:RemoveUser for adding and removing users.

ISC:RetrievePK and ISC:IsAddrOfUser for retrieving user addresses and verifying the legitimacy of their authorization.

ISC:AcquisitionUser is called by CSC during requests to obtain user-related evaluation criteria.

Table 4. Smart contract functions and descriptions.

Smart Contract	Function	Caller	Input	Output
ASC	RequestAccess	RE	$File_{ID}$, P_n, RE_{addr}, IN_{addr}	Call CSC: evaluaterrequest
	SaveDecision	SC: CSC	$Decision$	Save decision related to RE_{addr} and $File_{ID}$ in ASC
	RevokeAccess	/	RE_{addr}, $File_{ID}$	Revoke decision related to RE_{addr} and $File_{ID}$ within 15 minutes
	Verifydecision	PA	$File_{ID}$, RE_{addr}	Verify if RE_{addr} has decision permission for the target $File_{ID}$
CSC	Evaluaterrequest	SC: ASC	$File_{ID}$, RE_{addr}, IN_{addr}, R_n, $RE_{message}$	Call SSC, ISC, PSC. Then calculate request result and callback ASC: SaveDecision
SSC	Administution	GO	IN_{addr}	Add IN_{addr}
	RemoveInstitution	GO	IN_{addr}	Remove an institution
	RetrieveInstitution	SC: SSC	IN_{addr}	Check if IN_{addr} has been added
	IsAddrOfInstitution	GO	GO_{sig}	Verify if the regulator's address and GO_{sig} are valid
	AdministutionAttr	GO	IN_{addr}	Add attributes to the institution
	GetInstitutionAttr	GO	IN_{addr}	Get attributes of the institution
	Acquisition institution	SC: CSC	IN_{addr}	Get institution information
ISC	AddUser	IN	RE_{addr}	Add a user
	RemoveUser	IN	RE_{addr}	Remove a user
	RetrieveRE	SC: ISC	RE_{addr}	Search for the user address
	IsAddrOfUser	IN	IN_{sig}	Verify if RE_{addr} and IN_{sig} are valid
	AcquisitionUser	SC: CSC	RE_{addr}	Get user information
FSC	createPolicy	GO	P_n, R_n	Add a policy for a type of file
	ChangePolicy	GO	P_n, R_n	Change a policy for a type of file
	RemovePolicy	GO	$Policy_{id}$	Remove a policy for a type of file
	AcquisitionFilePolicy	SC: CSC	P_n, M	Get file policy
	AddFile	IN	$File_{ID}$, P_n, M	Add a file
	RemoveFile	IN	$File_{ID}$, P_n	Remove a file
	AcquisitionFileAttribute	SC: CSC	$File_{ID}$, M	Get file attributes

(5) Policy Smart Contract (FSC): The FSC defines, manages, and stores access policies. The policies in this contract are Access Behavior Analytical. It includes four functions:

FSC:CreatePolicy, FSC:ChangePolicy, FSC:RemovePolicy for creating, modifying, and deleting policies, respectively.

FSC:LoadPolicy for retrieving policies.

Only the GO can execute the functions in FSC to create, modify, or delete policies. Additionally, only CSC can call FSC:AcquisitionFilePolicy to retrieve policies for evaluation purposes.

4.5. Electronic medical record access control process

This section demonstrates the implementation of an access control framework for medical record sharing within a healthcare scenario using the proposed model. In the healthcare context, we have designed two distinct scenarios: the first scenario elaborates on the electronic medical record (EMR) storage process, while the second scenario provides a detailed description of the EMR access process.

4.5.1. Scenario 1

During a medical visit, the PA generates a public-private key pair, which is critical for ensuring end-to-end encrypted data transmission. Based on this, the storage scenario of EMRs was designed.

(1) When PA visits a healthcare institution with service capabilities, the public key is transmitted to the institution. The healthcare institution generates the patient's EMR data, computes a unique electronic data identifier using a hash algorithm, and encrypts the EMR data with the patient's public key.

(2) The healthcare institution assigns attributes to the EMR based on predefined file attributes set by the regulatory authority and forwards the data to AG.

(3) AG submits the encrypted file to ST and invokes the FSC: AddFile.

The process is illustrated in Figure 3.

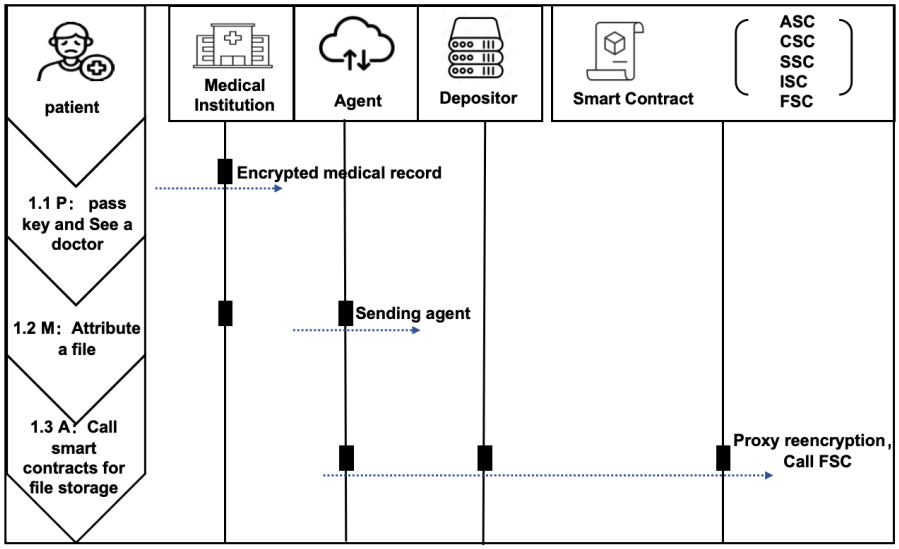


Figure 3. Electronic medical record storage process.

4.5.2. Scenario 2

In order to realize patient-centered access control, this paper uses proxy re-encryption technology to redesign the EMR access process. After the visitor identity and attributes are automatically verified through smart contracts, patients can view the audit information through the blockchain network, and finally determine whether to authorize the visitor. Make sure that the final visitor has access to EMR, as shown in Figure 4.

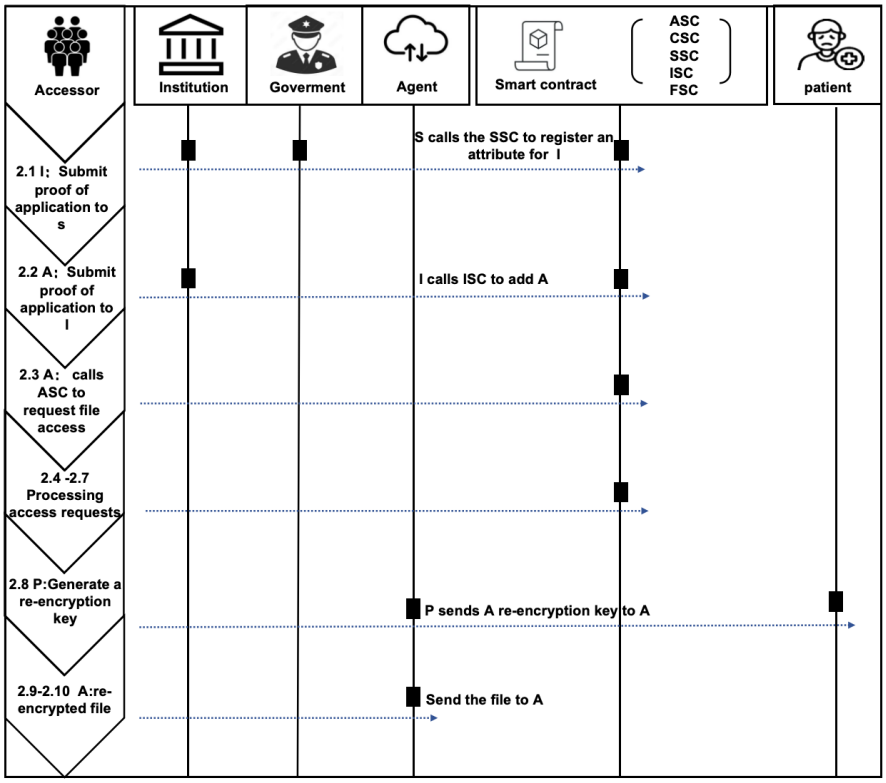


Figure 4. Electronic medical record access process.

(1) IN submits an identity verification request to GO, which reviews and registers. Upon approval, GO invokes the AddInstitution function in the Supervision Smart Contract (SSC) to add the authorized institution. Subsequently, GO calls the SSC:AddInstitutionAttr to assign specific attributes to the institution, determining the types of EMR files it is permitted to access.

(2) RE submits an identity verification request to their affiliated IN. After verifying the RE identity, IN invokes the ISC:AddUser to register RE.

(3) RE executes the Access Smart Contract (ASC) and awaits the processing result from the smart contract system.

(4) The ASC contract invokes CSC to perform the decision-making process.

(5) The CSC verifies whether IN is trusted by GO by querying the SSC contract. It then checks whether RE belongs to IN by querying the ISC contract. Finally, it retrieves file attributes from FSC:AcquisitionFilePolicy and verifies whether these attributes match the role of RE affiliated IN.

(6) The CSC completes the access policy evaluation and calls back ASC: SaveDecision to ST whether the RE's address has decision-making permissions for the target file ID. This decision aligns with the access policy tree.

(7) After the decision is generated, RE sends an access request to PA. PA executes the ASC:VerifyDecision to verify whether RE address has decision-making permissions for the target file ID.

(8) Upon successful verification, PA generates a re-encryption key using RE public key.

(9) PA sends the re-encryption key to AG which retrieves the file from ST provider and re-encrypts the encrypted data.

(10) AG transmits the re-encrypted data to RE, who decrypts it using their private key.

5. Performance evaluation

5.1. Security analysis

In this section, we perform a security analysis of the proposed large-scale medical data access control method, focusing on its resilience to attacks and its ability to protect data privacy.

(1) Data privacy protection and integrity: In the proposed framework, electronic medical record (EMR) data is protected throughout its lifecycle using proxy re-encryption technology. During data generation, the healthcare institution encrypts the EMR with the patient's public key. Since attackers do not possess the encryption key, they cannot initiate proxy re-encryption to decrypt the data, ensuring that data ownership remains with the data source. Furthermore, the integrity of the EMR is ensured by employing a hash algorithm during data generation. Any modification to the EMR would alter its hash value, making it impossible to match the original hash generated at the time of creation. Therefore, the proposed scheme effectively preserves data integrity.

(2) Resistance to spoofing attacks: Spoofing attacks are typically carried out by malicious actors attempting to gain unauthorized access to data. The proposed method resists such attacks by relying entirely on identity verification through smart contracts. When executing conditions within the smart contracts, only addresses explicitly permitted by the contract rules are recognized, preventing attackers from obtaining node keys. Moreover, the agent is limited to performing re-encryption operations and cannot access the specific content of the files. Additionally, since all EMRs stored in the cloud are encrypted, decryption is impossible without obtaining access permissions via the access control policy.

(3) Resistance to replay attacks: A replay attack occurs when an accessor resubmits a previously approved access request to the patient, or when an attacker intercepts and reuses an access request to gain unauthorized access to electronic medical records (EMRs). To mitigate such attacks, the Ethereum blockchain uses a nonce field and a time field. The nonce field records the position of the block, and the time field records the duration since the first block was created. If the nonce value of a transaction is less than that of the current block, or if the time field does not match the current timestamp, the transaction will not be executed, preventing replay attacks.

(4) Hash collision issues: In the proposed method, we employ the SHA-256 algorithm to hash electronic medical records (EMRs), generating a 256-bit hash value. Although hash collisions are theoretically possible, the probability of such an occurrence in this scheme is exceedingly low. Given a 256-bit hash length, the total number of possible hash outcomes is 2^{256}. Assuming the maximum number of EMRs accommodated in the system is $M = 10^9$, the probability of a hash collision can be calculated using the formula proposed in the literature:

$$P_{\text{collision}} \approx \frac{M^2}{2 \times 2^{256}} \tag{1}$$

Substituting the values into Equation (4–1), the probability of a hash collision in this scheme is calculated as 1.04×10^{-56}. Therefore, hash collisions are effectively impossible in the proposed EMR access control delegation scheme.

5.2. Experimental evaluation

To evaluate the proposed model,The experiment was conducted on an Ubuntu 22.04.2 LTS virtual machine running on VMware Workstation 16, with an Intel® Core™ i5-8300H CPU @ 2.30 GHz × 2, a 4-core processor, and 4 GB of RAM. Each machine was equipped with geth (Go Ethereum), and an Ethereum account was configured for each node, connecting all nodes to form a private Ethereum blockchain. Ethereum is an open-source platform with smart contract (scripting) capabilities, which was employed to implement the comprehensive access control functionality. The smart contracts were written in Solidity and compiled/deployed using Remix. On the patient side, web3.js was used to interact with the corresponding geth node via HTTP, monitoring the state of the Access Smart Contract (ASC). When an access request was approved, the system facilitated interaction with the agent.

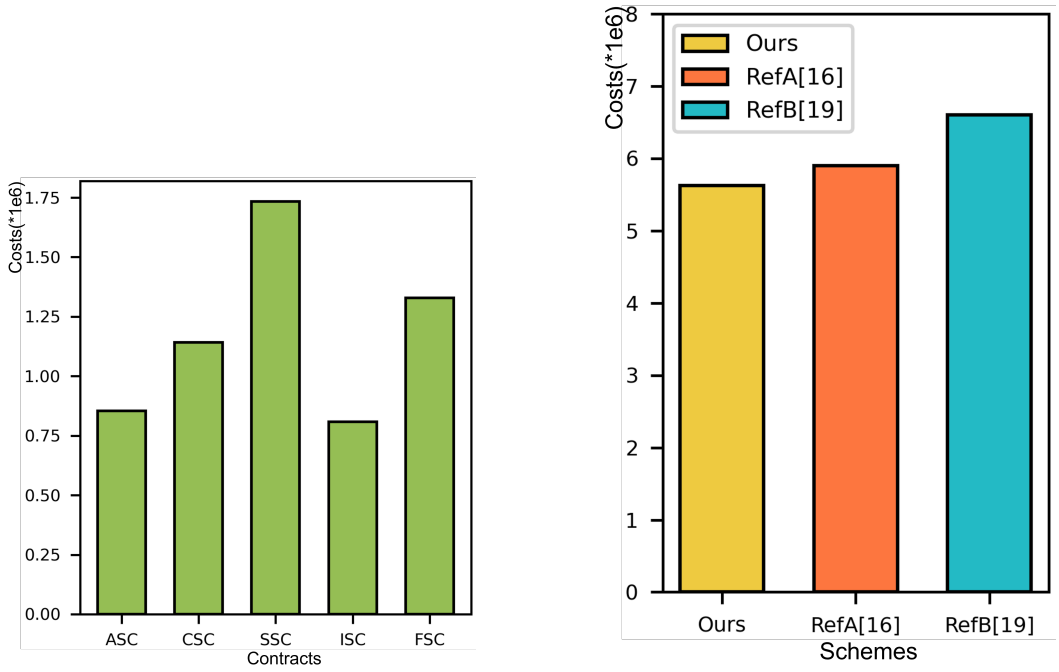
This section first evaluates the cost-effectiveness of the smart contracts in meeting the demands of large-scale access. Subsequently, it analyzes the encryption and decryption time of electronic medical records (EMRs) under large-scale access control. Finally, the efficiency of the proposed scheme is assessed and compared with other existing solutions. Both the proposed scheme and the comparative experiments were conducted under identical testing conditions.

5.2.1. Evaluation of smart contract access costs

In the proposed scheme, we utilize five distinct smart contracts to implement the access control framework for electronic medical records (EMRs). These contracts include the Access Smart Contract (ASC), Control Smart Contract (CSC), Supervision Smart Contract (SSC), Institution Smart Contract (ISC), and Policy Smart Contract (FSC). Each contract serves a specific function in the access control process, and their deployment costs are critical for evaluating the feasibility of the system in real-world applications.

The deployment costs for these contracts are illustrated in Figure 5. The ASC, responsible for executing access processes, incurs a deployment cost of 853,868 gas units. The CSC, which handles decision-making for access requests, requires 1,141,782 gas units. The SSC, tasked with managing authorized institutions, has the highest deployment cost at 1,732,104 gas units due to its complex role in regulatory oversight. The ISC, which manages users under institutional jurisdiction, costs 808,638 gas, while the FSC, which defines and stores access policies, requires 1,328,364 gas. The total deployment cost for the entire system is 5,864,756 gas .

Compared to the deployment costs reported in other studies [21,27], our scheme achieves a slight reduction in gas consumption while maintaining the capability to support large-scale, rapid, and convenient access. This efficiency is attributed to the optimized design of the smart contracts, which minimizes redundant computations and leverages Ethereum's gas-efficient operations.



(a) Smart contract gas fees **(b)** Cost comparison

Figure 5. Comparison of smart contract gas fees.

Additionally, the cost incurred by requesters when executing access requests aligns with real-world usage scenarios. Each access attempt requires a nominal gas fee, which serves as a deterrent against invalid or malicious access attempts. This economic mechanism ensures the honesty of requesters and mitigates the risk of system attacks, as each unauthorized or frivolous request would impose a financial cost on the requester.

5.2.2. Evaluation of encryption and decryption time

The timeliness of electronic medical record (EMR) sharing is a critical factor in healthcare applications, where delays can impact patient care and operational efficiency. To evaluate the performance of our model, we analyzed the impact of encryption and decryption latency, as well as proxy re-encryption, on the overall framework. This analysis ensures that the system can handle large-scale access requirements without compromising data security.

We collected medical records for 500 patients in real-world scenarios to simulate realistic conditions. The storage size data indicates that plain text files range from 10 KB to 300 KB, while medical record files containing images range from 1 MB to 30 MB. For uniformity, we conducted experimental evaluations with text files of sizes 50 KB, 100 KB, 200 KB, and 400 KB, and image files of sizes 1 MB, 3 MB, 10 MB, and 15 MB.

As shown in Figure 6(a) and Figure 6(b), the majority of the time is consumed by the encryption and decryption processes. For image files, the encryption and decryption times are higher, as expected, due to the larger file sizes. The results confirm that the proposed model can efficiently manage large-scale access requests while maintaining robust data security. The consistent performance of proxy re-encryption, in particular, ensures that the system can handle high-frequency access without introducing significant

latency. This makes the framework suitable for healthcare environments where timely access to medical records is critical.

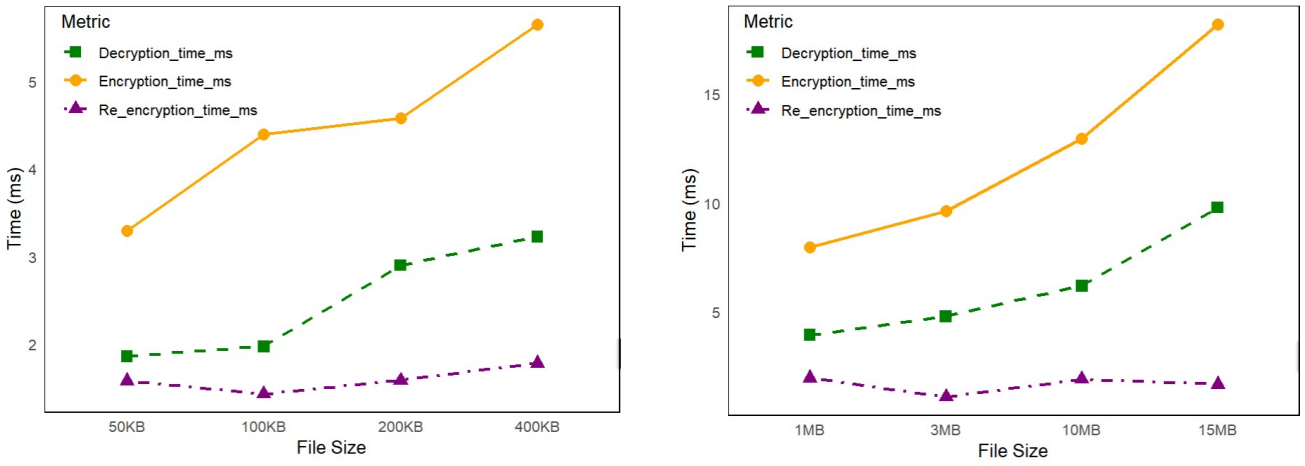


Figure 6. Proxy re-encryption encryption and decryption file time comparison. **(a)** Latency for encryption and decryption of EMR text files; **(b)** Latency for encryption and decryption of EMR images.

6. Conclusion

In this paper, we propose a large-scale access control model for electronic medical records (EMRs) based on Ethereum smart contracts. By utilizing smart contracts to evaluate access requests, the model addresses the issue of EMR ownership not being in the hands of patients in real-world scenarios. It offers a fast and efficient solution for large-scale access control, ensuring transparency and traceability in granting access permissions while maintaining end-to-end encrypted sharing of EMRs. The proposed scheme has been implemented on a private Ethereum blockchain, and experimental results show that the model effectively controls access requests while safeguarding the privacy and security of patient EMRs. Performance analysis confirms that the model meets practical cost requirements and significantly enhances access control efficiency.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (62141605, 62372493), the Beijing Natural Science Foundation (Z230001), the Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing (GJJ-23-001, GJJ-23-002) and the science and technology project of State Grid Corporation of China (5400-202255416A-2-0-ZN), and the National Natural Science Foundation of China under Grant, and National Key RD Program of China under Grant 2022ZD0116800, and the China Postdoctoral Fellowship Fund 2024M764092, and the Beihang Dare to Take Action Plan JKF-20240773.

Authors' contribution

Conceptualization, C.Z.; methodology, formal analysis, software, C.Z.; visualization, investigation, C.Z; data curation, software, C.Z., L.T.Y.; writing—original draft preparation, C.Z., Q.W.J.; writing—reviewing and editing, C.Z., H.J.C., L.T.Y., Z.Q.N., Q.W.J.; supervision, Z.Q.N., Q.W.J.; all authors have read and agreed to the published version of the manuscript.

Conflicts of interests

The authors declared that they have no conflicts of interests.

References

[1] Tian S, Yang W, Le Grange JM, Wang P, Huang W, *et al.* Smart healthcare: making medical care more intelligent. *Global Health J.* 2019, 3(3):62–65.

[2] Nowrozy R, Ahmed K, Kayes A, Wang H, McIntosh TR. Privacy preservation of electronic health records in the modern era: a systematic survey. *ACM Comput. Surv.* 2024, 56(8):1–37.

[3] Ullah F, He J, Zhu N, Wajahat A, Nazir A, *et al.* Ehr management evolution through purpose-based access control and blockchain smart contracts. *Int. J. Inf. Secur.* 2025, 24(1):63.

[4] Zuo C, Lin Z, Zhang Y. Why does your data leak? Uncovering the data leakage in cloud from mobile apps. In *2019 IEEE Symposium on Security and Privacy (SP)*, San Francisco, USA, May 19–23, 2019, pp. 1296–1310.

[5] Sandhu R, Munawer Q. How to do discretionary access control using roles. In *Proceedings of the third ACM workshop on Role-based access control*, Fairfax, USA, October 22–23, 1998, pp. 47–54.

[6] Hu VC, Kuhn DR, Ferraiolo DF, Voas J. Attribute-based access control. *Comput.* 2015, 48(2):85–88.

[7] Ferraiolo DF, Cugini JA, Kuhn DR. Role-based access control (RBAC): features and motivations. In *Proceedings of 11th annual computer security application conference*, New Orleans, USA, December 11–15, 1995, pp. 241–248.

[8] Barkley J. Comparing simple role based access control models and access control lists. In *Proceedings of the second ACM workshop on Role-based access control*, Fairfax, USA, November 6–7, 1997, pp. 127–132.

[9] Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008. Available: https://bitcoin.org/bitcoin.pdf (accessed on 25 February 2025).

[10] Buterin V. A next-generation smart contract and decentralized application platform. 2014. Available: https://ethereum.org/en/whitepaper (accessed on 25 February 2025).

[11] Zyskind G, Nathan O, Pentland AS. Decentralizing privacy: using blockchain to protect personal data. In *2015 IEEE security and privacy workshops*, San Jose, USA, May 21–22, 2015, pp. 180–184.

[12] Wang S, Zhang Y, Zhang Y. A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *Ieee Access* 2018, 6:38437–38450.

[13] Ouaddah A, Abou Elkalam A, Ait Ouahman A. FairAccess: a new Blockchain-based access control framework for the Internet of Things. *Secur. Commun. Netw.* 2016, 9(18):5943–5964.

[14] Putra DR, Anggorojati B, Hartono APP. Blockchain and smart-contract for scalable access control in Internet of Things. In *2019 International Conference on ICT for Smart Society (ICISS)*, Bandung, Indonesia, November 19–20, 2019, pp. 1–5.

[15] Ullah F, He J, Zhu N, Wajahat A, Nazir A, *et al.* Blockchain-enabled EHR access auditing: enhancing healthcare data security. *Heliyon* 2024, 10(16).

[16] Rashid A, Masood A, Khan AuR. ACS-IoT: smart contract and blockchain assisted framework for access control systems in IoT enterprise environment. *Wireless Pers. Commun.* 2024, 136(3):1331–1352.

[17] Tian J, Tian JF, Du RZ. MSLShard: an efficient sharding-based trust management framework for blockchain-empowered IoT access control. *J. Parallel Distrib. Comput.* 2024, 185:104795.

[18] Wang J, Li J. Blockchain and access control encryption-empowered IoT knowledge sharing for cloud-edge orchestrated personalized privacy-preserving federated learning. *Appl. Sci.* 2024, 14(5):1743.

[19] Tang F, Ma S, Xiang Y, Lin C. An efficient authentication scheme for blockchain-based electronic health records. *IEEE Access* 2019, 7:41678–41689.

[20] Jabbar R, Fetais N, Krichen M, Barkaoui K. Blockchain technology for healthcare: enhancing shared electronic health record interoperability and integrity. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, Doha, Qatar, February 02–05, 2020, pp. 310–317.

[21] Saini A, Zhu Q, Singh N, Xiang Y, Gao L, *et al.* A smart-contract-based access control framework for cloud smart healthcare system. *IEEE Internet Things J.* 2020, 8(7):5914–5925.

[22] Abid A, Cheikhrouhou S, Kallel S, Tari Z, Jmaiel M. A smart contract-based access control framework for smart healthcare systems. *Comput. J.* 2024, 67(2):407–422.

[23] Psarra E, Apostolou D, Verginadis Y, Patiniotakis I, Mentzas G. Permissioned blockchain network for proactive access control to electronic health records. *BMC Med. Inf. Decis. Making* 2024, 24(1):303.

[24] Ullah F, He J, Zhu N, Wajahat A, Nazir A, *et al.* Cost-effective EHR management: image compression and blockchain. *Int. J. Netw. Secur.* 2024, 26(5):874–884.

[25] Szabo N. Smart contracts: building blocks for digital markets. *Extropy* 1996, 18(2):28.

[26] Ateniese G, Benson K, Hohenberger S. Key-private proxy re-encryption. In *Topics in Cryptology-CT-RSA 2009: The Cryptographers' Track at the RSA Conference 2009*, San Francisco, USA, April 20–24, 2009, pp. 279–294.

[27] Wei L, Zhaoyang S, Wei S, Zhao T. Classification attribute access control method based on smart contract (In Chinese). *Appl. Res. Comput./Jisuanji Yingyong Yanjiu* 2022, 39(5).