

Review of Computer Engineering Research

2025 Vol. 12, No. 4, pp. 228-244

ISSN(e): 2410-9142

ISSN(p): 2412-4281


DOI: 10.18488/76.v12i4.4560

© 2025 Conscientia Beam. All Rights Reserved.



Blockchain-enabled secure EHR sharing with cloud storage using smart contracts and IPFS

 Sunitha B J^{1*}

 Saravana Kumar S²

^{1,2}School of Computer Science and Engineering, Presidency University, Bengaluru, India.

¹Email: sunithabj@presidencyuniversity.in

²Email: saravanakumar.s@presidencyuniversity.in



(+ Corresponding author)

ABSTRACT

Article History

Received: 27 June 2025

Revised: 23 September 2025

Accepted: 29 October 2025

Published: 2 December 2025

Keywords

Access control

Blockchain

Cloud computing

EHR

IPFS

Smart contracts.

The purpose of this study is to address the persistent security and privacy challenges in cloud-based Electronic Health Record (EHR) sharing by proposing a blockchain-enabled architecture that integrates decentralized storage and smart contracts. Traditional mobile cloud solutions improve data accessibility but rely on centralized control, making them vulnerable to unauthorized access, single points of failure, and limited patient transparency. To overcome these limitations, this research designs a user-centric access control framework that leverages the Ethereum blockchain, smart contracts, and the InterPlanetary File System (IPFS) within a mobile cloud environment. The methodology involves developing and deploying a prototype on Amazon Web Services, supported by an Android-based mobile application that enables healthcare providers and patients to interact with the blockchain network. Experimental evaluation was conducted using wearable sensor data to test the performance, scalability, and resilience of the proposed system. The findings indicate that the framework ensures secure EHR exchange, enforces fine-grained access policies, and achieves reduced latency compared to conventional centralized approaches. Unauthorized requests were reliably detected and blocked through the smart contract mechanism, while authorized users accessed records with minimal delay. The results also confirm the lightweight overhead of the system, making it practical for mobile healthcare environments. The practical implications of this work lie in offering a tamper-resistant, transparent, and patient-centric solution for medical data sharing, thereby improving trust, reducing administrative overhead, and supporting real-time healthcare services in distributed and resource-constrained settings.

Contribution/Originality: This study uniquely demonstrates a practical blockchain-enabled EHR sharing system by integrating Ethereum smart contracts with IPFS and deploying it on AWS with mobile devices. Unlike prior simulation-based works, it validates real-world feasibility, ensuring decentralized access control, reduced latency, and enhanced patient-centric privacy in mobile healthcare environments.

1. INTRODUCTION

In recent times, blockchain technology has gained substantial attention for its potential to improve medical and e-health services. Its decentralized architecture and inherent trustworthiness make it particularly well-suited for applications such as the secure exchange of Electronic Health Records (EHRs) and the regulation of data access among diverse healthcare institutions. Consequently, the integration of blockchain into healthcare systems is seen as a promising approach to enhancing service efficiency and driving innovation within the healthcare sector [1]. The integration of advanced technologies such as Mobile Cloud Computing (MCC) and the Internet of Medical Things

(IoMT) has further transformed e-health practices. Through mobile devices, including smartphones and wearable sensors, patients can collect health data at home and upload it to cloud platforms [2]. Healthcare professionals are then able to access this data in real-time, enabling prompt analysis and clinical support. This intelligent framework not only supports remote patient monitoring and ambulatory care but also reduces healthcare costs while improving service accessibility.

While cloud-based EHR storage offers these advantages, it also introduces major security challenges that hinder widespread adoption. A key issue is the secure exchange of EHRs between patients and healthcare professionals in mobile cloud environments. Without strong protection, unauthorized entities may gain access to sensitive data, compromising confidentiality, integrity, and trust. Additionally, patients often lack transparency and control over how their medical records are accessed or shared across different healthcare providers [3].

Conventional access control mechanisms typically assume cloud servers are fully trustworthy, placing them in charge of authentication and data management. However, in mobile cloud computing, servers are often "honest-but-curious" executing operations correctly while attempting to infer private information. This trust model introduces serious privacy concerns and increases the risk of data breaches. Furthermore, centralized access control systems create single points of failure, posing reliability and security risks for e-health infrastructures [4].

Blockchain-based access control offers a viable alternative by addressing these limitations. Its immutable ledger ensures data integrity, while its transparent and decentralized nature prevents unauthorized access and promotes accountability. By integrating smart contracts, blockchain enables autonomous and enforceable access control, reducing reliance on centralized authorities and enhancing system resilience and fairness. Motivated by these challenges and opportunities, this study proposes a novel blockchain-enabled framework for secure EHR sharing in mobile cloud environments. The core contribution lies in a user-centric access control mechanism supported by smart contracts and decentralized storage (IPFS), ensuring that only authorized entities can access EHRs while maintaining system scalability, integrity, and privacy [5].

Key Contributions of This Paper: This study contributes to the existing literature by presenting a novel EHR-sharing architecture that combines blockchain, IPFS, and smart contracts within a mobile cloud-based e-health framework. This study uses a new estimation methodology by implementing a custom data-sharing protocol and evaluating its performance and security through an Android-based AWS-integrated application. This paper reviews access control and blockchain applications in EHR sharing, outlines blockchain fundamentals, and presents the proposed system architecture. It details the prototype implementation using smart contracts and a custom data-sharing protocol, followed by experimental results, security and performance analysis, and concludes with key findings and future directions.

2. LITERATURE SURVEY

The secure sharing and access control of Electronic Health Records (EHRs) in cloud environments has garnered substantial research attention. Existing works propose a variety of mechanisms aimed at ensuring data confidentiality, integrity, and availability [6]. This literature review classifies the existing approaches into centralized and decentralized models, offering a critical comparison of their respective strengths and limitations.

1. **Centralized Approaches:** Early EHR sharing mechanisms often adopt a centralized architecture, relying on trusted intermediaries or centralized cloud servers to manage access control [7]. One such study proposed a broker-mediated access control mechanism, enabling selective sharing of EHRs among healthcare providers. While effective in simulation environments using Virtual Machines (VMs), the approach lacked validation on resource-constrained platforms such as smartphones, an important limitation for mobile health applications. To strengthen authentication, Public Key Infrastructure (PKI)-based solutions have been explored to uphold e-health security protocols within cloud platforms [8]. Although PKI facilitates secure communication, it often imposes additional complexity in terms of key management and computational overhead. Another conventional

approach involves the use of Ciphertext-Policy Attribute-Based Encryption (CP-ABE), incorporating an attribute authority responsible for issuing decryption keys based on predefined user attributes. This design provides fine-grained access control, allowing data owners to define specific access policies. However, the model operates within a centralized trust environment and overlooks the potential advantages of decentralized architectures. Moreover, most evaluations of such frameworks have remained simulation-based (e.g., on Ubuntu Linux systems), with limited focus on practical deployment and scalability.

2.1. Strengths of Centralized Approaches

1. Simpler architecture and implementation.
2. Efficient data retrieval and policy enforcement.

2.2. Limitations of Centralized Approaches

1. Single point of failure and increased vulnerability to insider threats.
2. Scalability and trust concerns in multi-institutional healthcare systems often lack patient-centric control and transparency.
3. Decentralized and blockchain-based approaches: To overcome the limitations of centralized systems, a growing body of literature has embraced decentralized architectures, particularly leveraging blockchain technology. Blockchain enables tamper-proof logging and decentralized trust, making it well-suited for healthcare scenarios involving multiple stakeholders. Studies have proposed blockchain-based EHR frameworks utilizing smart contracts to enforce access control policies. For instance, the MeDShare system facilitates data exchange among cloud service providers, incorporating monitoring features for access auditing and unauthorized access detection [9]. Despite these advances, most implementations focus on theoretical constructs or performance metrics such as network latency, without addressing data privacy risks or real-world deployment challenges. Some works propose storage optimization strategies through blockchain-integrated solutions, addressing issues related to the management of large-scale medical data [10-12]. These include hybrid models combining InterPlanetary File System (IPFS) with blockchain smart contracts to support secure data sharing in Internet of Things (IoT) and mobile environments. While these designs enhance auditability and resilience, they often do not resolve identity management and authentication issues, which remain critical for secure health data governance. Other researchers introduced user-oriented models focusing on identity protection and patient privacy, though they often lack comprehensive mechanisms for fine-grained access control and patient oversight [13].

2.3. Strengths of Decentralized Approaches

1. Enhanced transparency, immutability, and auditability.
2. Reduced reliance on central authorities.
3. Improved patient autonomy and control over personal health data.

2.4. Limitations of Decentralized Approaches

1. High computational cost and scalability concerns.
2. Limited practical deployment and real-world validation.
3. Incomplete identity and authentication mechanisms in many frameworks.
3. Hybrid and Emerging Architectures: Recent research trends highlight the use of hybrid architectures that combine centralized cloud storage with decentralized blockchain components. For example, integrating Ethereum, IPFS, and Attribute-Based Encryption (ABE) allows data owners to define and enforce complex access policies while leveraging decentralized storage to ensure availability and resilience. To improve search

efficiency in decentralized systems, smart contracts have also been employed for keyword-based retrieval, with implementations tested on Ethereum's Rinkeby network. These frameworks demonstrate improvements in data privacy and integrity, but challenges remain in optimizing gas costs, maintaining scalability, and integrating with existing healthcare infrastructure.

4. Identified Research Gaps: Despite the advancements, several critical challenges persist across the reviewed literature.
 - a. Overdependence on external key management and PKI in centralized models.
 - b. Lack of real-world deployment and validation in mobile or resource-limited environments.
 - c. Insufficient focus on patient-centric control and identity verification.
 - d. Limited exploration of fine-grained access control in dynamic mobile cloud contexts.
5. Motivation for This Study: Addressing the identified gaps, the present study proposes a blockchain-enabled data management framework optimized for mobile cloud-based e-health systems. The proposed architecture integrates IPFS for decentralized storage, Ethereum smart contracts for access enforcement, and mobile cloud infrastructure for enhanced accessibility. Unlike earlier work that is largely conceptual or simulation-based, this framework is practically implemented and deployed using mobile devices and Amazon-hosted blockchain services.

The key contributions include:

- a. Design of a holistic architecture combining blockchain and smart contracts for secure EHR sharing tailored for mobile health environments.
- b. Practical deployment and validation, emphasizing real-world performance metrics such as access control efficiency, latency, and system integrity.
- c. Comprehensive security evaluation and discussion of unresolved integration challenges, contributing actionable insights for future healthcare blockchain implementations.

3. TECHNICAL BACKGROUND

This study employs the Ethereum blockchain platform as the foundation for developing the proposed e-health system. As a decentralized blockchain network, Ethereum offers functionality similar to Bitcoin but stands out due to its versatility and programmability, which support the creation of diverse blockchain-based applications, including those in the healthcare sector. We explore the essential features of the Ethereum network that are integral to the architecture and functionality of our system. Like Bitcoin, Ethereum operates on a blockchain structure composed of blocks that contain both transaction records and smart contracts. These blocks are verified and appended to the blockchain using the proof-of-work consensus mechanism, whereby miners perform computational tasks to maintain network security and consistency. Each block references the hash of its predecessor, preserving the sequential integrity of the chain. Through cryptographic hashing, Ethereum ensures that the data within the blockchain remains tamper-proof and immutable, preventing unauthorized alterations.

- 1) Ethereum Accounts: The Ethereum network distinguishes between two categories of accounts: externally owned accounts (EOAs) and contract accounts. Every account is assigned a unique 20-byte address and is linked to a cryptographic key pair consisting of a public and a private key. To engage with the Ethereum blockchain, users are required to create an account, which serves as their identity within the decentralized network.
- 2) Smart Contract: A smart contract is an autonomous computer program that executes automatically when predefined conditions are satisfied. On the Ethereum blockchain, a smart contract functions as a special type of account that stores both data and executable code with various programmable functions. Users can interact with smart contracts using their Ethereum accounts through application binary interfaces (ABI). Functions

within a smart contract can be invoked by initiating a new transaction from an account. This capability enables entities to perform tasks such as data transmission, request processing, and access control.

- 3) Transaction: In the Ethereum network, a transaction is a structured data unit that facilitates the transfer of Ether, the platform's native cryptocurrency, between accounts. Each transaction typically comprises several elements: account nonce, recipient address, gas price, gas limit, Ether amount, sender's digital signature, and an optional data field. A sample transaction can be expressed as: Tx = new Transaction (Sender address, gas price, gas limit, recipient address, amount, payload data). Within the proposed framework, the data field is specifically employed to include request identifiers, which are used to trigger data access operations through smart contracts on the cloud storage system.

4. SYSTEM MODEL

This section outlines the system architecture and introduces the concepts of data uploading and sharing within our system. Additionally, it highlights the design goals discussed in this paper.

4.1. System Architecture

We propose an e-health model on a mobile cloud platform where patient records, collected via local gateways, are stored in a public cloud and shared with healthcare providers (Figure 1). These EHRs, containing personal and medical data, are linked to each patient through a unique Patient ID (PID) and classified by Area ID (AID). The wearable sensor network is assumed to be privately managed by the patient, with data collected via a smartphone-integrated mobile app. A patient's blockchain address is defined as $\text{Addr} = \{\text{AID}, \text{PID}\}$. Due to storage limitations on the blockchain, only patient addresses are recorded on-chain, while the actual EHRs are stored in decentralized cloud storage. A cloud-based EHR manager (ME) is introduced for record handling. Accessing a patient's health data requires knowledge of their blockchain address. The overall system data flow is illustrated in Figure 2.

We assume that each healthcare provider (e.g., doctor, physician, or nurse) is equipped with a mobile device and assigned a unique Healthcare Provider ID (HPID), which is used to authenticate and retrieve EHRs from cloud storage. Through this mechanism, providers can access patient medical histories for clinical analysis and diagnostic purposes, facilitating informed medical decision-making.

To enable secure and decentralized EHR sharing, we implement a cloud-integrated blockchain network. Ethereum is selected as the smart contract platform due to its programmability, decentralization, and established ecosystem, as discussed in the previous section. The primary components of the proposed cloud-blockchain architecture are detailed as follows.

EHRs Manager: Within the proposed data sharing architecture, the EHRs manager plays a vital role by supervising user transactions across the blockchain network. Its responsibilities include managing data storage initiated by mobile gateways and regulating data access requests from mobile users. This oversight is implemented through smart contracts, which ensure strict adherence to predefined user access policies.

Administrator: The administrator is tasked with overseeing transaction and operational control within the cloud environment, including the assignment, alteration, or revocation of access rights. This entity is also responsible for deploying smart contracts and holds exclusive authority to update or revise the access control policies embedded within them.

Smart Contracts: Smart contracts outline the authorized operations within the access control framework. Users communicate with these contracts via their unique contract address and the Application Binary Interface (ABI). They are designed to detect, verify, and process access requests, enabling permissions for medical users by triggering specific transactions or messages. Since all participants in the blockchain network can interact with the smart contract and invoke its functions, it serves as a fundamental element of the proposed e-health system.

Decentralized Storage: Since storing and sharing large datasets directly on the blockchain is inefficient, this work employs the InterPlanetary File System (IPFS) a decentralized, peer-to-peer storage solution as an effective approach for blockchain-based data sharing. IPFS is built on foundational technologies such as the BitTorrent protocol and the Kademlia Distributed Hash Table (DHT). By removing dependence on centralized servers, IPFS distributes data across multiple nodes within a unified file system. This architecture offers key advantages over conventional cloud storage, including the elimination of single points of failure, enhanced storage scalability, and more efficient data retrieval. In IPFS, each file is uniquely identified and retrieved through the cryptographic hash of its content.

In the proposed framework for EHR sharing, medical records are encrypted and distributed across IPFS nodes, while their corresponding content hashes are managed by the EHRs manager and indexed within the Distributed Hash Table (DHT). To strengthen decentralized storage and enable controlled data exchange, IPFS is integrated with smart contracts, allowing for fine-grained user access control. The EHR storage system deployed over IPFS is organized as a network of independent nodes $S=(s_1,s_2,\dots,s_n)$, with each node responsible for storing EHRs corresponding to patients from a designated geographic region, as depicted in Figure 3.

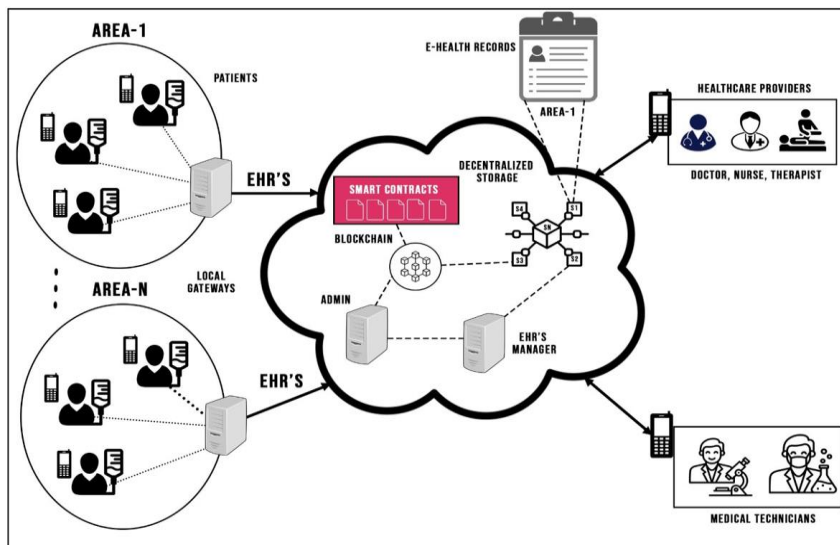


Figure 1. Architecture of a blockchain-enabled E-health system in a mobile cloud environment.

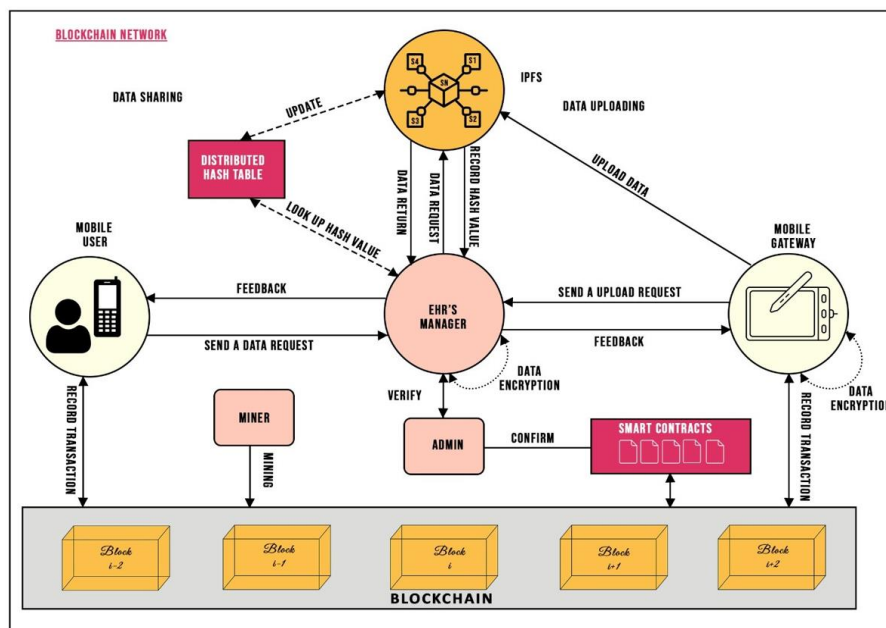


Figure 2. Proposed system data flow for blockchain-integrated mobile cloud platform.

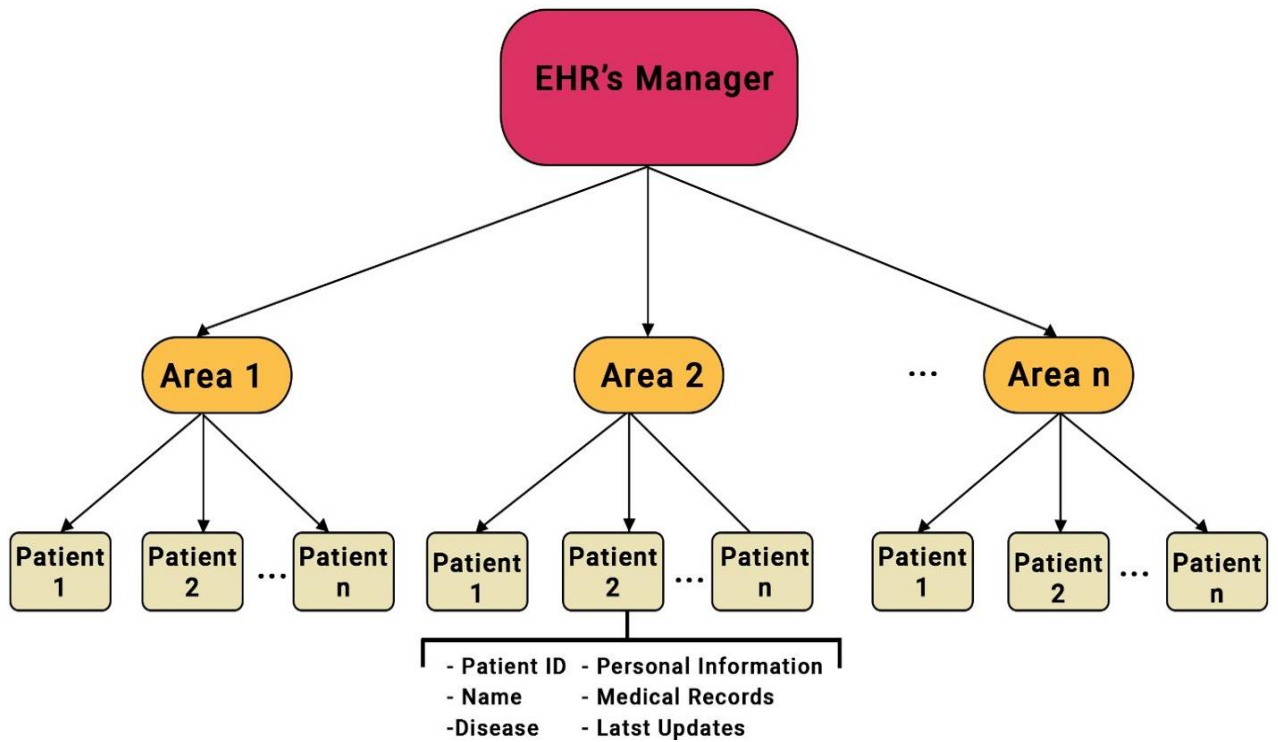


Figure 3. Architecture of the cloud-based EHR storage system.

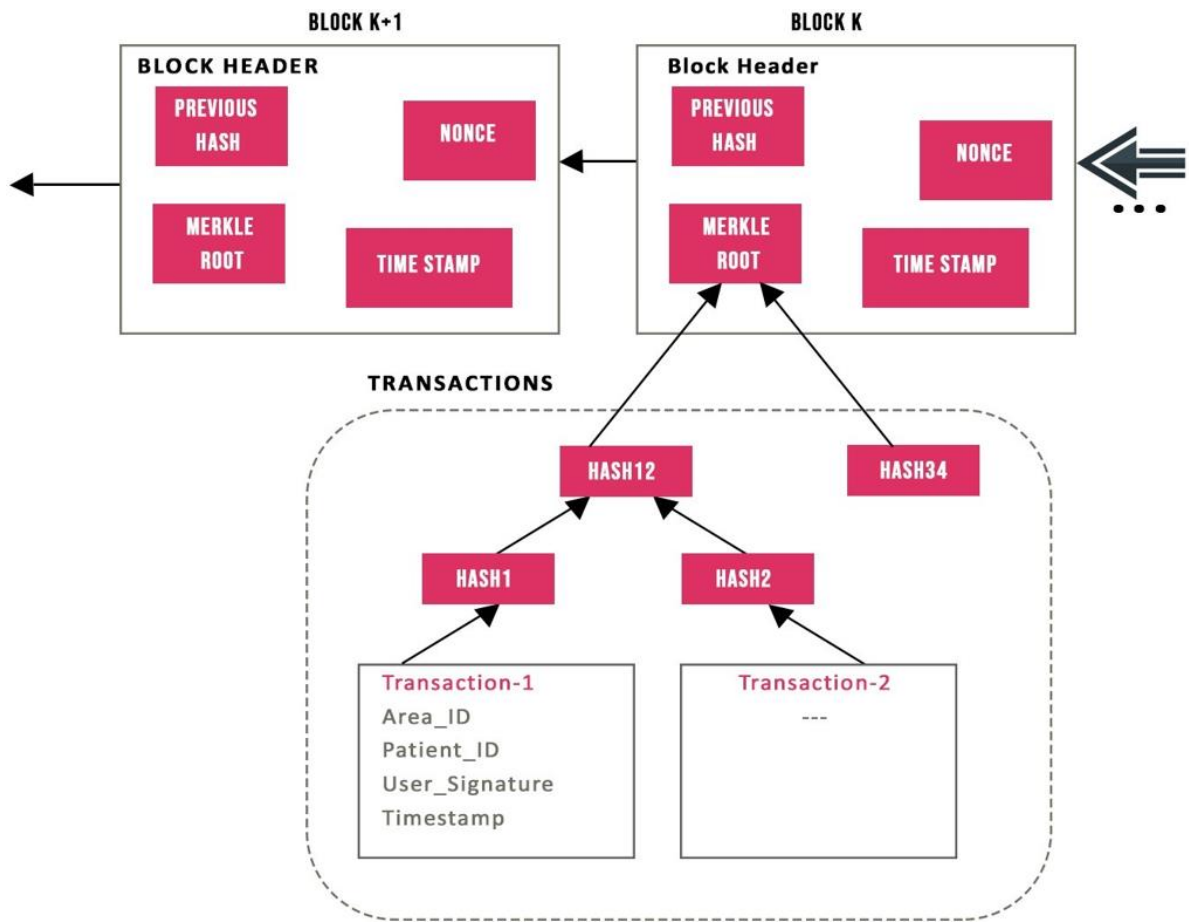


Figure 4. Structure of the data block.

Figure 5 Illustrates Data Uploading and Sharing over a Mobile Cloud-Blockchain System.

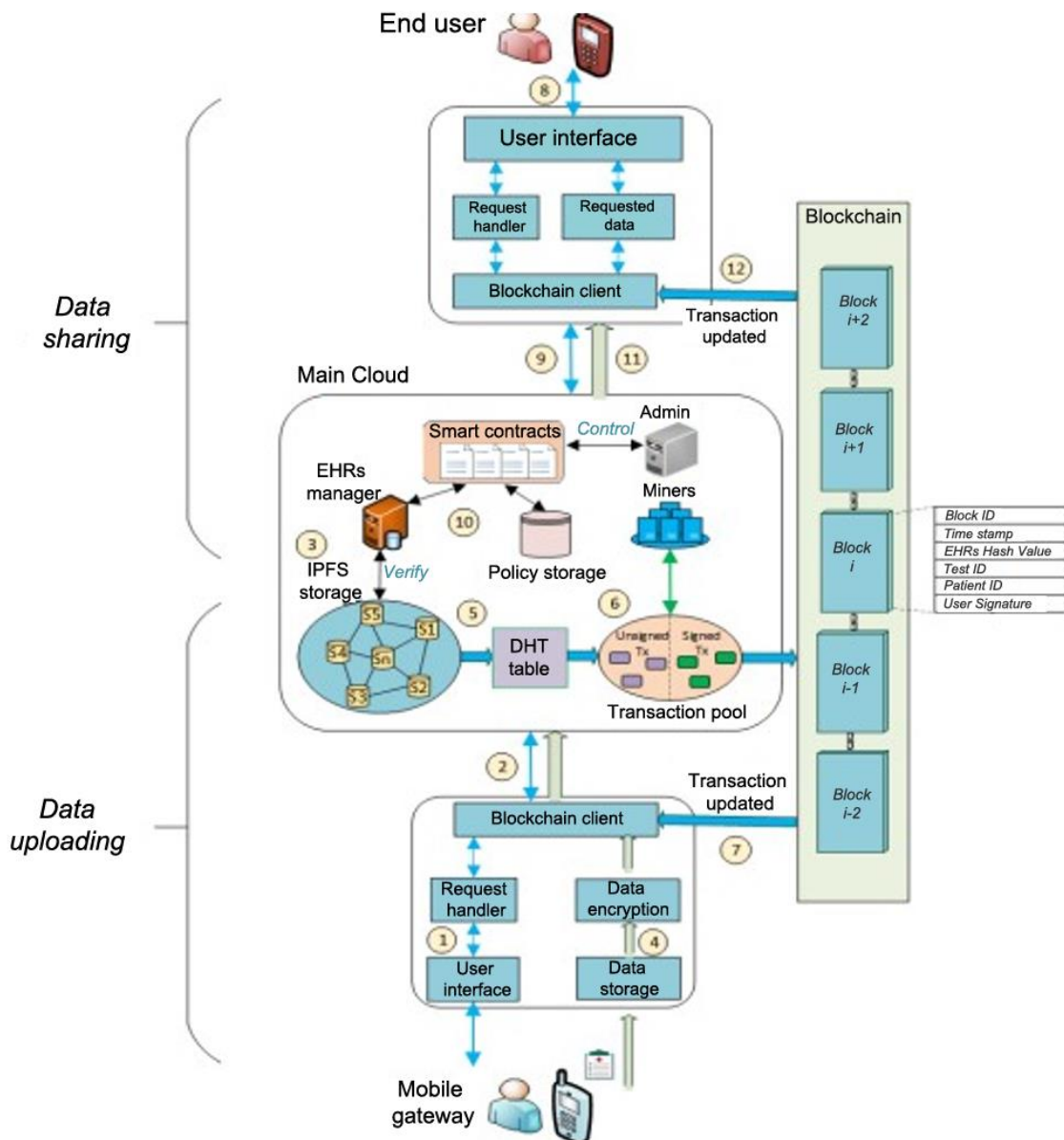


Figure 5. Workflow of data upload and sharing over a mobile cloud-blockchain system.

Figure 6 Illustrates the Distributed Hash Table (DHT) in the Proposed IPFS Storage Design.

Area ID	Patient ID	Hash value of data file
A101	P1001	Qm690ebb015f1c9d9...
	P1002	Qm40b3643d2facf894...
	P1003	Qm435039fc72817c61...

A201	P2001	Qmd6f71b01b9f987b5...
	P2002	Qm06bcab5ff755d822...
	P3003	Qm861c8f5ec34e4ba3...

...

Figure 6. Architecture of the distributed hash table (DHT) in the proposed IPFS storage design.

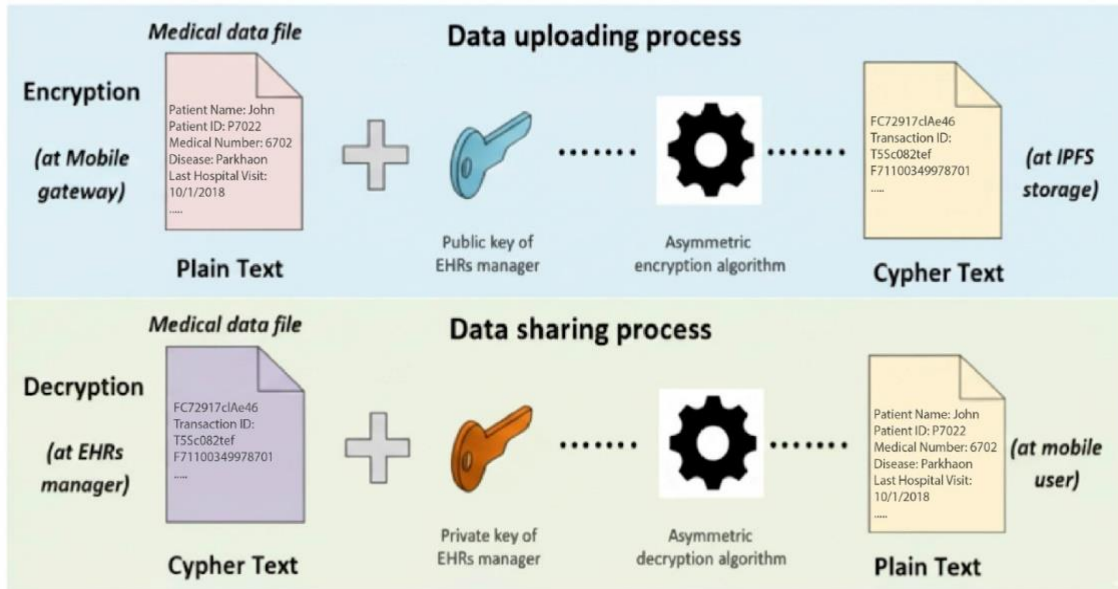


Figure 8. Experiment setup for EHRs sharing simulation.

Algorithm: Authorization and Retrieval Process for Blockchain-Enabled EHR Access.

Input: Transaction Tx.

Output: Access Decision Result.

- 1) Initialization phase (executed by EHRs manager).
- 2) Accept the incoming transaction Tx from the mobile client.
- 3) Extract the sender's public key: $PK \leftarrow Tx.get\ Sender\ Public\ Key()$.
- 4) Transmit the public key PK to the administrative authority.
(Msg. Sender = EHRs Manager)
- 5) Authentication and validation phase (handled by smart contract and admin).
- 6) Check if the public key PK exists in the access control registry: Policy List (PK) $\leftarrow true$.
- 7) End if.
- 8) Decode the transaction content: $decoded\ Tx \leftarrow Abi\ Decoder.Decode\ Method(Tx)$.
- 9) Retrieve the request parameters from the data payload.
- 10) $Addr \leftarrow web3.eth.getData(Decoded\ Tx[Data\ Index])$.
- 11) Extract identifiers for Area and Patient.
- 12) $AID \leftarrow Addr[Index[AID]]$; $PID \leftarrow Addr[Index[PID]]$.
- 13) EHR Retrieval Procedure (Managed by Smart Contract).
- 14) Repeat.
- 15) If policy list (AID) = True then.
- 16) If policy list (PID) = True then.
- 17) Grant access and retrieve EHR data.
- 18) $Result \leftarrow Retrieve\ EHRs(PK, Addr)$ (see Algorithm 4).
- 19) Exit loop.
- 20) Else.
- 21) Impose an access restriction penalty.
- 22) $Result \leftarrow Penalty(PK, action)$.
- 23) Exit loop.
- 24) Else.
- 25) Apply access violation penalty.
- 26) $Result \leftarrow Penalty(PK, action)$.
- 27) Exit loop.
- 28) End if.
- 29) Continue until the transaction is resolved.
- 30) Return the final access outcome Result.

Data block structure: Figure 4 illustrates the structure of an EHR block, comprising the following key components.

Transaction Records: In the proposed framework, transaction data is organized using a Merkle tree structure, where each leaf node represents a data access request initiated by a mobile user. To initiate a transaction, the user must input specific patient details such as Area ID and Patient ID, which are then embedded into the transaction. This transaction is digitally signed using the user's private key and timestamped, ensuring authenticity and establishing a trusted interaction between the user and the cloud server.

Block Header: The block header contains essential metadata required for validating the corresponding data block.

Hash: To maintain data integrity, the block's SHA-256 hash is utilized. As shown in Figure 4, the hash value is computed by combining the hashes of individual transactions, for example:

$$Hash_{12} = Hash(Hash_1 + Hash_2) = Hash[(Tx_1.Hash) + (Tx_2.Hash)].$$

Previous hash: The hash of the preceding block, which is utilized for validating the current block.

Merkle Root: A data structure designed to store a set of transactions within each block.

Nonce: It represents a nonce value computed through the proof-of-work algorithm executed by miner nodes, with the objective of generating a block hash that satisfies the network's target difficulty threshold.

Timestamp: It indicates the timestamp marking the creation of the block and also corresponds to the time of the final transaction recorded within the block.

5. IMPLEMENTATION

To assess the feasibility and effectiveness of the proposed system, we developed a proof-of-concept (PoC) implementation of a decentralized access control mechanism for secure EHR sharing within mobile cloud environments. The implementation details, along with the system configuration parameters, are outlined in the following subsections.

5.1. System Settings

Here is a rephrased and more formal version of your paragraph, tailored for a research paper: We designed and implemented an EHR sharing framework over a mobile cloud environment, as illustrated in [Figure 8](#). A private Ethereum blockchain network was deployed on Amazon Web Services (AWS), utilizing two AWS EC2 virtual machines configured as mining nodes. Additionally, two Ubuntu 16.04 LTS virtual machines were designated for administrative control and EHR management, respectively. The smart contract, developed using the Solidity programming language, was deployed through AWS Lambda functions. These functions interact with the blockchain via the Web3.js API.

Mobile users interact with the smart contracts through Android smartphones, each configured with a Geth client, a command-line interface implemented in Go, which enables the devices to function as Ethereum nodes. Through the Geth client, users can create Ethereum accounts and interact directly with the blockchain for secure data access. Furthermore, the Web3.js library, a lightweight JavaScript library for blockchain interaction, was employed to develop the mobile application for seamless communication with the Ethereum network. For experimental evaluation, we used two Sony smartphones running Android OS version 8.0 to analyze the performance and practicality of the proposed EHR sharing system.

Subsequently, we configured the InterPlanetary File System (IPFS) on the Amazon Web Services (AWS) platform to establish a decentralized storage infrastructure. IPFS is currently compatible with cloud-based deployment on AWS, and the system configuration is illustrated in [Figure 9](#). In this study, medical records were managed using our mobile healthcare platform, *BioKin*, which comprises a network of wearable sensors and an Android-based mobile application designed to collect and process patient data for cloud storage. A dedicated Java library was integrated into the mobile application to enable seamless uploading of data to the IPFS cloud environment.

For testing purposes, we assumed that medical records were collected via BioKin sensors and subsequently stored in the IPFS network. To ensure the confidentiality of medical data at the mobile gateway, we implemented an asymmetric encryption scheme using the RSA algorithm, developed in Java, and integrated into the Android application.

During the data upload phase, all files were encrypted using the public key of the EHR manager. In the data sharing phase, decryption was performed using the manager's corresponding private key, thereby maintaining secure asymmetric encryption, as depicted in [Figure 7](#).

```

AWS::CloudFormation::Init
config:
  files:
    '/tmp/IPFS_install.sh':
      content: |
        #!/bin/bash -e
        sudo yum install wget -y;
        wget https://dist.ipfs.io/go-ipfs/v0.4.17/go-ipfs_v0.4.17_linux-amd64.tar.gz;
        tar -xvzf go-ipfs_v0.4.17_linux-amd64.tar.gz;
        cd /go-ipfs
        sudo ./install.sh;
        ipfs init;
        ipfs config Addresses.API /ip4/0.0.0.0/tcp/5001
        ipfs config --json API.HTTPHeaders.Access-Control-Allow-Origin ['*']
        ipfs config --json API.HTTPHeaders.Access-Control-Allow-Methods ['GET',
*POST*]
        ipfs config --json API.HTTPHeaders.Access-Control-Allow-Headers
['Authorization']
        ipfs config --json API.HTTPHeaders.Access-Control-Expose-Headers ['Location']
        ipfs config --json API.HTTPHeaders.Access-Control-Allow-Credentials ['true']
        ipfs daemon &
        sleep 10
        ipfs bootstrap rm --all
        exit
      mode: '000731'
      owner: root
      group: root
  commands:
    install_ipfs:
      command: /tmp/IPFS_install.sh

```

Figure 9. Configuration setup for deploying and integrating the InterPlanetary File System (IPFS) within the AWS cloud infrastructure.

5.2. Access Control for EHRS Sharing

In this subsection, we develop a smart contract to implement the proposed access control model. Additionally, we design an access protocol that defines the operational workflow of the EHR sharing scheme.

1) Smart Contract Design: We begin by implementing a data-sharing smart contract, managed by the system administrator, to oversee transaction operations within the blockchain network. In this context, PK represents the user's public key, user role indicates the user's role, and Addr denotes the patient's blockchain address. The contract is designed to support five core functions, described as follows.

Add User (PK, user role) (executed by Admin): This function enables the registration of a new user within the main contract. Each user is identified by their public key and assigned a corresponding role based on their request. Additionally, user information is stored in the cloud as part of the system's database.

Delete user (PK, user role) (executed by Admin): This function facilitates the removal of users from the network using their associated public key. Upon removal, all corresponding personal data is also erased from the cloud storage.

Policy list (PK) (executed by Admin): A healthcare provider and patient can establish a mutual access policy that defines their relationship within the medical service context. For instance, a patient may designate a specific doctor as their sole caregiver, granting exclusive access to the patient's EHRs. The policy list stores the public keys of all authorized entities, enabling the smart contract to verify identities during transaction processing.

2) DATA SHARING PROTOCOL: This study introduces an access protocol specifically designed to enable robust user access control within the electronic health record (EHR) sharing system. The protocol's execution is

triggered by a mobile user's initiation of an EHR request to the cloud and proceeds through four distinct, sequential steps.

- Step 1: Transaction Pre-Processing (Executed by the EHRs Manager): Upon receiving a new transaction Tx from a mobile user (e.g., a healthcare provider or patient), the EHR manager extracts the requester's public key using the 'Tx. Get Sender Public Key()' function. This public key is then forwarded to the smart contract for validation.
- Step 2: The administrator initiates the verification process upon receiving a transaction containing the user's public key (PK) from the EHR manager (where msg.sender equals ME). The administrator verifies the requester's access rights by comparing the PK against the policy list within the smart contract. If the PK is authorized, the request is approved, and data access permission is granted. Conversely, an unauthorized PK triggers the smart contract's penalty() function, penalizing the requester. This action subsequently blocks all EHR access attempts and rejects the request from the blockchain network.
- Step 3: EHRs Retrieval (Executed by the Admin): Once access permission for the transaction is granted, the smart contract decodes the transaction using the 'ABI Decoder: Decode Method (Tx)' function to extract the address information of the EHRs from the transaction's data field (refer to Section II). This allows the administrator to identify the corresponding Area ID and Patient ID. Subsequently, the administrator forwards a request to the EHR manager to locate and retrieve the relevant medical records from the decentralized cloud storage system.
- Step 4: EHRs Feedback (Executed by the EHRs Manager): After locating the requested EHRs, the EHR manager delivers the data to the requester through an off-chain communication channel between the cloud and the mobile user. Following the completion of this exchange, a new blockchain transaction containing primarily the patient's address metadata is generated and broadcast to all network participants. Since only minimal metadata is stored on-chain, this approach ensures low storage overhead, thereby enhancing the scalability and practical applicability of the proposed blockchain-based e-health system. The complete procedure of the access protocol is detailed in Algorithm 1.

5.3. Testing Methodology

Based on the configured hardware and software environment, this section presents a comprehensive evaluation of the proposed EHR sharing system through real-world testing. An Ethereum blockchain network was deployed on Amazon Web Services (AWS) and integrated with a blockchain client application accessible via Android smartphones. For evaluation purposes, the experimental setup focused on a specific testing zone (AreaID = 1), comprising ten patients (PatientID = 1 to 10). Accordingly, a single IPFS storage node was configured to store ten distinct medical data files corresponding to each patient. These data files were collected through wearable sensors, as illustrated in Figure 8.

Two primary use cases were considered to assess access control.

- 1) An authorized user, such as a medical professional, who functions as a legitimate participant in the blockchain network and is permitted to query patient data.
- 2) An unauthorized user, modeled as a potential adversary attempting to gain illicit access to sensitive medical records stored on the blockchain-integrated cloud platform. The performance evaluation concentrated on two key metrics: access control effectiveness and network overhead. To evaluate access control, a mobile cloud application was developed and deployed on the Ethereum blockchain. Both authorized and unauthorized users interacted with the system through the blockchain client on Android devices, initiating data requests via smart contract transactions. Access control efficacy was assessed based on the system's ability to permit legitimate users to efficiently retrieve EHRs while identifying and mitigating unauthorized access attempts. The smart contract-driven access control mechanism is expected to provide reliable protection against data breaches,

ensuring secure and trustworthy EHR sharing within the blockchain environment. In addition to access control, network performance, particularly latency, was evaluated to determine the system’s suitability for real-time EHR sharing. Authentication time, executed via smart contracts, was measured using Amazon’s Kinesis Data Analytics service. Furthermore, communication latency was assessed by recording the time required to transmit a data request and receive a response on the blockchain client application. The results of these evaluations, based on the described testing methodology, are presented in the subsequent section.

6. EXPERIMENTAL RESULTS

To realize the proposed EHRs sharing framework, we initiated the deployment of a private Ethereum blockchain on Amazon Web Services (AWS) as depicted in Figure 11. All data access events and transaction records are captured and displayed via a web-based monitoring interface. Following the blockchain configuration, we implemented smart contracts, integrated IPFS for decentralized storage, configured network components, and linked the system with mobile applications to construct the complete e-health ecosystem. The framework was then operated and assessed based on two key performance indicators: access control effectiveness and network overhead.

```
pragma solidity ^0.4.24;
contract EHRs {
    address admin;
    mapping (address => bool) dataOwner;
    constructor() public {
        admin = msg.sender; // Sets deployer as admin
    }

    modifier onlyAdmin { // Restricts access to admin
        require(msg.sender == admin);
        _;
    }

    function addDataOwner(address newOwner) public onlyAdmin returns (bool success) {
        dataOwner[newOwner] = true; // Admin adds new user
        return true;
    }

    function readData(address requester) public constant returns (bool hasPermission) {
        // Checks if user has permission
        return dataOwner[requester];
    }
}
```

Figure 10. Smart contract code for EHR sharing.

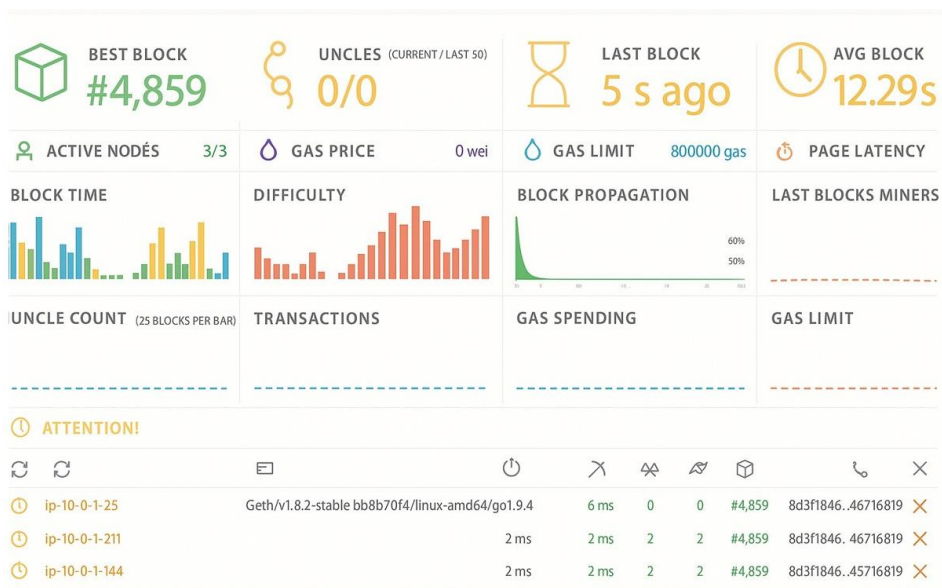


Figure 11. The Ethereum blockchain deployed on AWS to facilitate EHR sharing.

A. Access control performance: To evaluate the effectiveness of our EHR sharing framework with the implemented access control mechanism, we demonstrate two use cases involving authorized and unauthorized access scenarios (as illustrated in Figure 12). The primary goal of our system is to enable verified entities such as healthcare providers to securely and efficiently access electronic health records (EHRs) stored on the cloud, while simultaneously preventing unauthorized access to sensitive data. In the authorized access scenario, a mobile user (e.g., a doctor) employs the mobile application's interface to create an Ethereum account and register the necessary credentials for blockchain interaction (Figure 12(a)).

Upon approval by the cloud-based EHR manager, the user initiates a transaction to retrieve a patient's EHRs by specifying the corresponding AreaID and PatientID (Figure 12(b)). The system processes the request and returns the relevant medical data, which is displayed on the user's mobile interface. This allows healthcare providers to review and analyze the patient's medical history and deliver appropriate care. Following this, the transaction is appended to the blockchain by a designated miner and broadcast across the network. This ensures that the patient remains informed about every access to their EHRs, thereby reinforcing individual data ownership and network transparency (Figure 12(c)).

For unauthorized access attempts, the smart contract mechanism enforces predefined access policies to validate user permissions. Any invalid or malicious request is automatically rejected by the access control protocol, triggering a warning notification to the requester (Figure 12(d)). Additionally, the event is logged on the blockchain as a transaction denoting unauthorized access (Figure 12(e)). This smart contract-based enforcement of access policies effectively addresses previously identified challenges in maintaining control over patient data and tracking user interactions, thereby enhancing both system dependability and privacy protection. The evaluated results demonstrate that our access control framework provides secure data sharing among mobile users, supports robust identity verification and authentication, and safeguards sensitive medical information from external threats. These outcomes validate the attainment of our first design objective outlined in Section IV-D.

B. Network latency: We evaluated the average time required by the cloud to handle multiple simultaneous access requests (Figure 13(a)). The smart contract-based user authentication mechanism introduced additional processing time compared to a non-authenticated approach, primarily due to identity verification and access authorization procedures. However, even in the worst-case scenario involving seven concurrent requests, the overhead remained minimal, approximately 100 ms, indicating that the added latency is negligible and acceptable for practical applications. This demonstrates the lightweight nature of our access control and authentication design.

Furthermore, we assessed the communication overhead involved in accessing EHRs stored in the blockchain-enabled IPFS cloud storage system (Figure 13(b)). We measured the end-to-end delay from initiating a data access request to receiving the EHR data on a mobile device. A comparative analysis was conducted between our decentralized blockchain storage (IPFS) and a traditional centralized solution using AWS S3. Results across varying EHR file sizes indicate that our blockchain-based model introduces lower time overhead than the centralized approach. This confirms the efficiency of our decentralized EHR sharing mechanism. In terms of network latency, the results affirm that our system delivers lightweight EHR sharing capabilities with minimal system overhead, outperforming conventional cloud-based solutions. These findings validate the second design objective outlined in Section IV-D.



Figure 12. Visualizing EHR access outcomes on Android devices: 12.a) Registering a user with an Ethereum account, 12.b) EHRs access results of an authorized user, 12.c) Log of approved EHR access transactions, 12.d) EHR results obtained by an unauthorized user, and e) Record of unauthorized access to EHRs.

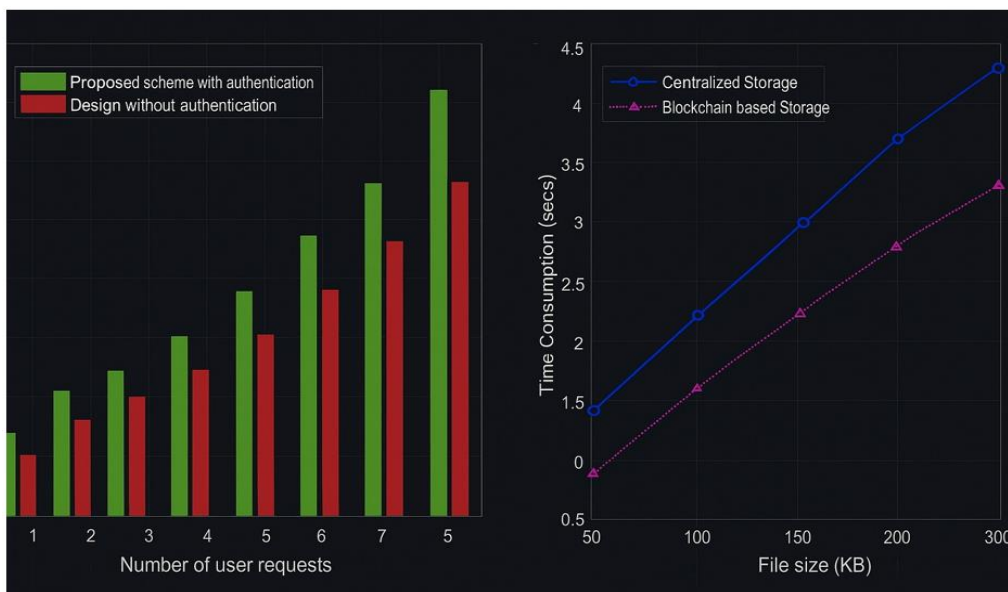


Figure 13. Time costs: a) Comparing cloud processing time with user request latency, and b) Performance timing for accessing EHRs from the cloud.

7. CONCLUSIONS

The proposed blockchain-enabled data management framework significantly enhances both the privacy and performance of EHR sharing in mobile cloud-based e-health systems. By integrating decentralized storage (IPFS), smart contracts, and fine-grained access control, the system ensures secure and efficient exchange of medical data among healthcare stakeholders. The practical deployment on mobile infrastructure further demonstrates its real-world viability and technical effectiveness. Looking forward, future work will focus on integrating machine learning-based anomaly detection for proactive threat monitoring and extending the framework to support interoperability across heterogeneous healthcare platforms. From a practical perspective, the system empowers healthcare providers with a tamper-resistant, transparent, and patient-centric approach to medical data access, potentially reducing administrative overhead, improving trust in data sharing, and enhancing patient outcomes in distributed care environments.

Funding: This study received no specific financial support.
Institutional Review Board Statement: Not applicable.
Transparency: The authors state that the manuscript is honest, truthful, and transparent, that no key aspects of the investigation have been omitted, and that any differences from the study as planned have been clarified. This study followed all writing ethics.
Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: Both authors contributed equally to the conception and design of the study. Both authors have read and agreed to the published version of the manuscript.

REFERENCES

- [1] C. Patel and N. Doshi, "Secure lightweight key exchange using ECC for user-gateway paradigm," *IEEE Transactions on Computers*, vol. 70, no. 11, pp. 1789-1803, 2021. <https://doi.org/10.1109/TC.2020.3026027>
- [2] I. Cherkaoui, "Diffie-Hellman multi-challenge using a new lossy trapdoor function construction," *IAENG International Journal of Applied Mathematics*, vol. 51, no. 3, pp. 736-742, 2021.
- [3] M. M. Ulla and D. S. Sakkari, "Research on elliptic curve crypto system with Bitcoin curves - SECP256k1, NIST256p, NIST521p and LLL," *Journal of Cyber Security and Mobility*, vol. 12, no. 1, pp. 103-128, 2023. <https://doi.org/10.13052/jcsm2245-1439.1215>
- [4] M. A. Khan, M. T. Quasim, N. S. Alghamdi, and M. Y. Khan, "A secure framework for authentication and encryption using improved ECC for IoT-based medical sensor data," *IEEE Access*, vol. 8, pp. 52018-52027, 2020. <https://doi.org/10.1109/ACCESS.2020.2980739>
- [5] Y. Park and J. Park, "Analysis of the upper bound on the complexity of LLL Algorithm," *Journal of the Korean Society for Industrial and Applied Mathematics*, vol. 20, no. 2, pp. 107-121, 2016. <https://doi.org/10.12941/jksiam.2016.20.107>
- [6] M. M. Ulla, Preethi, M. S. Khan, and D. S. Sakkari, "Implementation of elliptic curve cryptosystem with Bitcoin curves on SECP256k1, NIST256p, NIST521p, and LLL," *Journal of ICT Standardization*, vol. 11, no. 4, pp. 329-354, 2023. <https://doi.org/10.13052/jicts2245-800X.1141>
- [7] D. Boneh and R. Venkatesan, "Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes," presented at the Annual International Cryptology Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996.
- [8] J. Breitner and N. Heninger, "Biased nonce sense: Lattice attacks against weak ECDSA signatures in cryptocurrencies," presented at the International Conference on Financial Cryptography and Data Security. Cham: Springer International Publishing, 2019.
- [9] D. Kumari, A. S. Parmar, H. S. Goyal, K. Mishra, and S. Panda, "HealthRec-chain: Patient-centric blockchain enabled IPFS for privacy preserving scalable health data," *Computer Networks*, vol. 241, p. 110223, 2024. <https://doi.org/10.1016/j.comnet.2024.110223>
- [10] Z. Sun, D. Han, D. Li, X. Wang, C.-C. Chang, and Z. Wu, "A blockchain-based secure storage scheme for medical information," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, p. 40, 2022. <https://doi.org/10.1186/s13638-022-02122-6>
- [11] A. Shahzad, W. Chen, Y. Zhang, and R. Kumar, "Zero-trust medical image sharing: A secure and decentralized approach using blockchain and the IPFS," *Symmetry*, vol. 17, no. 4, p. 551, 2025. <https://doi.org/10.3390/sym17040551>
- [12] S. Luo, N. Han, T. Hu, and Y. Qian, "Secure sharing of electronic medical records based on blockchain," *International Journal of Distributed Sensor Networks*, vol. 2024, no. 1, p. 5569121, 2024. <https://doi.org/10.1155/2024/5569121>
- [13] G. Lax, R. Nardone, and A. Russo, "Enabling secure health information sharing among healthcare organizations by public blockchain," *Multimedia Tools and Applications*, vol. 83, no. 24, pp. 64795-64811, 2024. <https://doi.org/10.1007/s11042-024-18181-4>

Views and opinions expressed in this article are the views and opinions of the author(s), Review of Computer Engineering Research shall not be responsible or answerable for any loss, damage or liability etc. caused in relation to/arising out of the use of the content.