Alexandria Engineering Journal (2023) 68, 205–226

HOSTED BY

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

ORIGINAL ARTICLE

# Blockchain based efficient tamper-proof EHR storage for decentralized cloud-assisted storage

**Dharavath Ramesh** [a], **Rahul Mishra** [a], **Pradeep K. Atrey** [b], **Damodar Reddy Edla** [c], **Sanjay Misra** [d,*], **Lianyong Qi** [e]

[a] *Indian Institute of Technology (ISM), Dhanbad 826004, India*
[b] *College of Engineering and Applied Sciences, University at Albany, State University of New York (SUNY), Albany, NY 12222, USA*
[c] *National Institute of Technology, Farmagudi, Goa 403401, India*
[d] *Department of Computer Science and Communication, Østfold University College, Halden, Norway*
[e] *College of Computer Science and Technology, China University of Petroleum (East China), China*

Received 14 October 2022; revised 2 December 2022; accepted 7 January 2023
Available online 20 January 2023

**KEYWORDS**

Blockchain;
Electronic Healthcare Record (EHR);
InterPlanetary File System (IPFS);
Cloud storage;
Timeliness

**Abstract** In recent years, the rapid and wide-ranging implementation of a cloud-based electronic healthcare record (EHR) storage system has shown significant advantages in effectively managing EHR for healthcare organizations and patients. However, in the cloud-based EHR storage model, the patients no longer have direct control of their EHR, whereas healthcare organizations may access the outsourced EHR whenever necessary. It may always cause severe security issues, specifically when healthcare organizations collude with the cloud service provider (CSP) to conceal any medical malpractice. Therefore, to deal with these significant concerns, we have introduced a novel blockchain based efficient tamper-proof model for EHR storage in decentralized InterPlanetary File System (IPFS) storage in the cloud - "TAC-EHR". The key idea of the model is that every operation involves outsourcing EHRs and integrating these EHRs into a transaction on the public blockchain provides computationally unforgeability to the outsourced EHRs. Moreover, the proposed EHR storage model can also manage batch outsourcing, i.e., numerous EHR outsourcing for multiple patients by multiple doctors simultaneously, in an effective manner. The experimental and security analysis demonstrates that the proposed blockchain based cloud-assisted EHR storage model efficiently assures intractability computationally and outperforms the existing models in terms of computational and communication overhead.



* Corresponding author.
E-mail addresses: drramesh@iitism.ac.in (D. Ramesh), rahul.18DR0107@cse.iitism.ac.in (R. Mishra), patrey@albany.edu (P.K. Atrey), dr.reddy@nitgoa.ac.in (D.R. Edla), sanjay.misra@hiof.no (S. Misra), lianyongqi@gmail.com (L. Qi).

Peer review under responsibility of Faculty of Engineering, Alexandria University.

## 1. Introduction

In recent years, cloud computing has extensively been used in various fields. Cloud computing refers to delivering various

https://doi.org/10.1016/j.aej.2023.01.012
1110-0168 © 2023 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Engineering, Alexandria University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

services over the internet, i.e., storage, networking, and software [1]. Cloud storage plays a key role in cloud computing through directly transferring data and access control to the cloud, which reduces the development cost and eliminates the need to manage the storage infrastructure on its own. The term "Electronic Health Record" (EHR) refers to a computer-based database that stores many types of medical data, including but not limited to patient identification information, medical history, diagnoses, allergies, vital signs, lab test findings, and more. This information is useful for medical personnel to monitor the patient's condition and plan treatment strategies. Moreover, these days, incorporating cloud services into electronic health systems has shown significant potential, and a wide range of unheard-of advantages to managing electronic healthcare records (EHRs) [2,3]. These cloud based EHR storage models facilitate healthcare institutions to seamlessly manage EHRs via cloud storage services and contribute significantly to dispute settlements in medical malpractices. Although these models make significant benefits more alluring than ever, EHR outsourcing has created severe security concerns. As an EHR is sensitive data, it should never be leaked for confidentiality reasons. However, these traditional cloud-based EHR storage models would not take accountability for EHR's privacy preservation against adversaries in their Service Level Agreements; but only commit to secure the privacy to the extent possible [4]. Moreover, in these existing models, the cloud service providers (CSPs), being third-party servers, are typically considered semi-malicious entities. Being a semi-malicious entity, the CSPs provide the storage service to the end-users. However, it can access, modify and leak data to others for its benefits [5,6]. These security issues mainly encompass EHR confidentiality, integrity, and availability. We briefly discuss these as follows:

1. *EHR confidentiality*: CSP can leak the patient's EHR and identity information for personal benefits, causing privacy loss to the patient.
2. *EHR integrity*: Since patients have no direct control over the outsourced EHRs in CSP, the integrity of the outsourced EHRs is always at high risk. Further, whenever any type of medical conflict arises, there is no way for patients to verify whether or not the outsourced EHRs have been modified.
3. *EHR availability*: Apart from these, cloud computing and storage are considered as a centralized system, so whenever the cloud servers fail or break down, all the patients or doctors would be severely affected.

We further elaborate on the above three issues. In a traditional cloud-based EHR storage system, initially, doctors generate the patient's EHR and outsource it to the CSP, where EHRs are signed and encrypted by the doctors [7]. So, the patients have no role in signing and encrypting the EHRs. Thus, maintaining the integrity of the outsourced EHRs at CSP is also challenging when doctors outsource the generated EHRs on patients' behalf. Since, in many real-world adversarial situations, doctors can be semi-malicious. Doctors can be trustworthy during the diagnosing period, but they can become malicious afterward. So, they can modify, forge, or delete the outsourced EHRs for personal benefits or hide their medical malpractices. Moreover, the centralized cloud storage can also causes the single point of failure on any failure at CSP.

Apart from these, there is a strong assumption in the existing model [8] that the CSP will never collude with the malicious doctor to modify the outsourced EHRs. This assumption does not hold in situations where the doctors can incentivize the CSP to tamper with the outsourced EHRs, then such type of misdeed is hard to detect. Therefore, to resist the collusion between rational CSP and malicious doctors, a trusted server is introduced to authenticate all the doctors. However, the computationally intractability of such methods depends on the reliability and unforgeability of trusted servers. Specifically, it is challenging to resist collusion between any two entities without any trusted third-party server being introduced.

In this paper, we propose a blockchain-based tamper-proof efficient EHR storage model for the cloud environment called - TAC-EHR. It assures the integrity and proper confidentiality of the outsourced EHRs from unauthorized modification with a trusted third-party entity. Moreover, the computational unforgeability of the proposed model is also assured even if any two entities collude, i.e., malicious doctor and the rational CSP. The primary idea is to employ blockchain technology, which offers tamper-proofing and a decentralized way for conducting transactions without any central authority being introduced. In the proposed model, the generated EHRs are integrated into a transaction on the permissionless or public blockchain, i.e., Ethereum blockchain [37,38]. The public blockchain is used as an open, distributed, and tamper-proof ledger to account for the possession of the token values or records all transactions between any two parties involved. Due to the computational immutability of the public blockchain, i.e., Ethereum blockchain, once a transaction is recorded in the blockchain, it cannot be modified and deleted. However, it is not reasonable to integrate the complete EHR on the blockchain, as the total resource needs for every node would be incredibly high on the blockchain. So, due to the limited storage space of every block on the blockchain, we employ a distributed file system, i.e., InterPlanetary File System (IPFS), which follows a content-based addressable method to store and search the data with high integrity and durability [9]. In IPFS, there is no centralized server; also, the EHRs are distributed and stored on various IPFS nodes throughout the internet. Therefore, IPFS technology doesn't have a single point of failure, ensuring the proper EHR availability to doctors and patients.

Furthermore, IPFS facilitates the distribution of massive amounts of EHRs with high efficiency [10]. Each of the EHRs stored on the IPFS system have a distinct hash-string, by which anyone can easily access the stored EHR in the distributed IPFS-nodes. In the proposed methodology, whole EHRs are stored in IPFS based distributed storage model. Thus, only the hash-string of EHRs are needed to integrate into transactions on the blockchain and also used to map the whole EHR in the IPFS based distributed storage. Therefore, the proposed model endorses large-scale EHR and has excellent robustness. Moreover, the proposed methodology also deploys an efficient and secure password-based key-agreement method to create secure communication between patients, hospitals, and doctors. This secure password-based key-agreement method can resist the vulnerable password guessing attacks, maintaining a more substantial security guarantee. To this end, we propose a cost-effective and more secure cloud-based EHR storage model via blockchain with improved security assurances and overall efficiency.

### 1.1. Our contributions

The main contributions of this manuscript are summarized as follow:

1. We employ a novel EHR storage architecture based on blockchain. This assures the confidentiality and integrity of the outsourced EHRs with a trusted entity, where the EHRs are integrated into immutable transactions on the blockchain. This considerably simplifies system administration and eliminates the need to implement new cryptographic systems to meet varying levels of security risk.
2. Also, we use decentralized storage (IPFS), which assigns a unique hash to each outsourced EHRs, preventing repetitive storage while reducing space. It also allocates distributed storage to outsourced EHRs to avoid a single point of failure, ensuring the proper availability of outsourced EHRs.
3. We also enhance the proposed model to support batch-outsourcing, storage and integrating multiple generated EHRs into a transaction on the blockchain, where multiple requests for the above-discussed tasks can be efficiently performed simultaneously.
4. The proposed model separates the transactions for publishing the outsourced EHRs from transactions for fine-grained access control. Simultaneously, the outsourced EHRs are stored on various IPFS nodes, effectively reducing the computation and communication overhead.
5. We also provide the formal security analysis to ensure the proper computational intractability of the proposed model. Even if any two entities collude, the malicious entities with a limited fraction of the network's total computational power cannot break the validity of blockchain and the infeasibility of the model. Moreover, the performance analysis shows the efficiency and practicability of the proposed model.

### 1.2. Organization

The rest of the paper is organized as follows; Section 2 introduces some previously designed frameworks on cloud-based secure EHR storage. Section 3 describes the preliminaries used throughout the paper. In Section 4, we describe the system model, the adversary model, and some other related definitions with design objectives. In Section 5, we discuss the complete model of proposed blockchain-based secure EHR storage in the cloud-environment. Detailed security analysis of the proposed model is described in Section 6. In Section 7, we analyze the performance of the proposed model in terms of communication and computation overheads. Section 8. Finally, Section 9 concludes this proposed model.

## 2. Related work

### 2.1. Traditional cloud-assisted EHR storage model

A cloud-based EHR storage system provides the users (i.e., individual patients or healthcare organizations) an effective and convenient way to regulate their sensible EHRs. As the EHRs contain the most confidential and crucial details for any patients [11], cloud-based EHR storage system often faces the challenges regarding the privacy and security of the outsourced EHRs [12–15]. Therefore, to attain proper security for patient's privacy towards internal adversaries, the whole EHRs are outsourced in encrypted form. Earlier, a large number of methodologies proposed to ensure the proper integrity and privacy for outsourced EHRs in a cloud model. Lee et al. [16] designed a key-management based effective model to secure EHRs. This methodology establishes a trusted third-party server to manage the patient's secret keys. However, due to some consequences, this trusted third-party server may be interested in accessing the information from EHRs, which causes some serious concerns related to the privacy of the patients. Sun et al. [17] drafted a secure EHR storage model to assure proper infeasibility for patient's privacy without any trusted third-party server being introduced. Nevertheless, the system model of this scheme does not match the basic cloud-based storage system requirements. Generally, in these discussed models, the EHRs are outsourced by the patients, whereas the doctors always required to send the generated EHR to the particular patients. Thus, in these models, the patients have a heavy burden for computation and communication overhead. Moreover, to attain the required security for the outsourced EHRs on semi-trusted CSP, attribute-based encryption (ABE) is implemented to ensure fine-grained access control, which also conceals the values of the distinct and responsive attributes in the access policy. Zhang et al. [18] reviewed these schemes and observed that the CSP contains a significant quantity of duplicate EHR. So, to reduce the storage overhead at CSP, the authors [18] designed an efficient model to enable the CSP to eliminate the duplicate EHRs, which effectively reduces the storage overheads. Wei et al. [19] introduced a CPABE scheme-based secure EHR-sharing model in the cloud paradigm. Their proposed model attained the user-revocation, EHR confidentiality, and key delegation of users. Nevertheless, in this model, user revocation is periodically executed to ensure security against internal threats from malicious users. This model also employed the less expressive LSSS access structure. However, these methodologies include secure and efficient storage for EHRs in the CSP. In contrast, some severe issues still exist in these methodologies, i.e., how to avoid the internal adversaries and failures or breakdown at CSP.

### 2.2. Blockchain based EHR storage model

Recently, some authors introduced the blockchain-based model for EHR storage in the cloud-environment [20–24]. Yu et al. [25] proposed an attribute-based signature model with various authorities to ensure the integrity of EHRs. After diagnosis, the patient's treatment-data is integrated into one single block. However, healthcare data can be massive and relational, which requires broad-range searching. Huang et al. [26] proposed a patient-centric blockchain-based model - "HealthBlock", to support the computational infeasibility against modification attacks on stored EHRs. So, the patients with the original password can only access the EHRs from the blockchain. They have designed a proof-chain to store manipulation logs of patients' stored EHRs. Later, the proof-chain logs were utilized as proof to protect rights. Furthermore,

HealthBlock is also susceptible to offline-dictionary and replay attacks. In 2021, Li et al. [27] proposed a blockchain-based EHR storage model with an attribute-based model to ensure proper security with privacy preservation. However, EHRs are directly stored on the blockchain at CSP in this model, leading to a new centralization concern. Simultaneously, Li et al. [28] compiled a blockchain-assisted EHR storage model with an effective provenance method. This model employed the public blockchain to secure the provenance of EHR records. So, this model efficiently audited the EHRs and securely maintained the corresponding provenance. However, due to the limited storage in blocks on the blockchain, increasing the size of EHRs or the number of EHRs, these models can lead to unacceptable authentication delays.

The authors Hussien et al. [29] proposed a "smart contract-based searchable attribute-based encryption (ABE) - SC-ABE" for secure EHR storage over cloud paradigm by integrating smart contract, CPABE, searchable symmetric encryption, and IPFS storage. Dagher et al. [30] recently proposed an efficient model, where hash-values of outsourced EHRs are stored in blocks on the blockchain while forwarding the specific query link information on transactions through HTTPs. So, this model also suffers from a severe Denial of Service (DoS) attack. Benil et al. [31] emphasized that scalability of medical records along with secure storage is a major concern for Blockchain-based EHR systems. These researchers created a test case in which smart contracts are used to improve data scalability. Furthermore, in real-scenario, doctors can only be fully trustworthy during the diagnosing-period. However, if any malicious doctors incentivize the CSP to modify the outsourced EHRs, it becomes challenging to disclose such misbehavior by malicious doctors [32]. Recently, Yang et al. [48] proposed an attribute-based outsourcing decryption model, which significantly reduces the computational overhead of users. Also, the ABE and ABS schemes improve the computational efficiency of the model. However, the consortium blockchain framework leads to a new centralization concern with severe key-escrow issues for secure EHR storage. Scheme [49] introduced IPFS based decentralized EHR storage model with permissioned blockchain to attain proper availability of EHR in cloud paradigm. However, this scheme still has some limitations, such as authorized access, timeliness of EHRs, and the functional challenges with the stored data in the blockchain. Chelladurai et al. [50] proposed a decentralized blockchain structure with modified merkle hash tree (MMHT) to share healthcare records. The proposed model relied on novel MMHT with efficient hashing procedure to secure verification and validation of all transactions for EHR sharing. Therefore, we propose a secure and efficient blockchain-based methodology for EHR storage in the cloud environment, which follows the timeliness of EHRs and assures the proper integrity of outsourced EHRs. Table 1 illustrates a comparative analysis of the existing cloud-based EHR storage models and the proposed model.

## 3. Preliminaries

In this section, we analyze some preliminaries to support the proposed methodology. We also give a brief explanation of bilinear pairing, blockchain, and some other related definitions.

### 3.1. Bilinear pairing and complexity assumptions

#### 3.1.1. Bilinear-pairing

Let $E_n$ be an elliptic curve defined over $F_q, n \in Z_p^*$ with gcd($n$, $p$) $=1$. Now, assume $(G_1, G_2)$ be the two multiplicative cyclic group of prime order $p$ and g is the fixed generator of $G_1$. So, a map functions as $e : G_1 \times G_1 \rightarrow G_2$ is described as cryptographic bilinear map iff it satisfies the following properties.

1. Non-Degeneracy: For any $a, b \in G_1, a \neq b$ then $e(a, b) \neq 1$.
2. Bilinearity: For any $\forall r, s \in Z_p$ then $\forall a, b \in G_1, e(aR, bS) = e(aR, S)^b = e(R, bS)^a = e(R, S)^{ab}$
3. Computability: For any $a, b \in G_1$ there exists an efficient algorithm to calculate $e(a, b)$.

**Definition 1: Decisional Diffie-Hellman (DDH) problem:** Let $G_1$ be a multiplicative cyclic group of prime order $p$ with $g$ as generator of $G_1$, and randomly select $(a, b, c) \in Z_p$, given $(g^a, g^b, g^{ab})$. Finally, determine whether $g^c = g^{ab}$.

**Definition 2: Decisional Diffie-Hellman (DDH) assumption:** Let $G_1$ be a multiplicative cyclic group of prime order $p$ with $g$ as generator of $G_1$, and randomly selects $(a, b, c) \in Z_p$. Then, the DDH-assumption is that it is hard to differentiate between the distribution of $(g^a, g^b, g^{ab})$ and $(g^a, g^b, g^c)$. More explicitly, we describe $Advt_{G_1}^{DDH}$ as follows;

$$Advt_{G_1}^{DDH}(A) = |Pr[A(g, g^a, g^b, g^{ab}) = 1]$$

The hard DDH-assumption holds in $G_1$ iff $Advt_{G_1}^{DDH}(A)$ is negligible for any probabilistic polynomial time adversary $A$.

**Definition 3: Computational Diffie–Hellman (CDH) assumption:** Let $G_1$ be a multiplicative cyclic group of prime order $p$ with $g$ as generator of $G_1$, and randomly selects $(a, b, c) \in Z_p$, given $(g, g^a, g^b)$. Then, it is computationally intractable to calculate $\Gamma = g^{ab}$. More explicitly, we describe $A_{G_1}^{CDH}$ as follows;

$$P[A_{G_1}^{CDH}(g, g^a, g^b \in G_1) \rightarrow g^{ab} \in G_1 : \forall_{a,b \in_R Z_p}] \leqslant \in$$

The hard CDH assumption holds in $G_1$ iff $A_{G_1}^{CDH}$ is negligible for any probabilistic polynomial time adversary $A$.

### 3.2. InterPlanetary file system (IPFS)

IPFS is an open-source, peer-to-peer distributed hypermedia protocol that aims to function as a ubiquitous file system for all computing devices [9,10]. It uses content-based addressing to uniquely identify the stored data-files in the global namespace, by assigning distinct hash-values for each data-file. Thus, the users can locate and access their respective data-files by relying on the hash-values of their respective stored data-flies. It maintains a good deduplication model without the restriction of any centralized server. In the IPFS network, for any frequently requested data-file, it forms a duplicate-data along the request-path. So, on the next request, it can be directly accessed locally. IPFS distributed storage system

**Table 1** Comparative analysis of different schemes and proposed model.

| Schemes | Proposed methodologies | Privacy preserving | Security against severe issues (forgery, modification, data-deduplication, key- escrow, reply and replace attacks) | Timeliness of EHR | Blockchain framework | Protection against single point of failure for storage |
|---|---|---|---|---|---|---|
| Sun et al. [17] | SSE (encryption technique) with ID-based bilinear pairing | ✓ | X | X | X | X |
| Zhang et al. [18] | Attribute based access control policy technique | ✓ | X | X | X | X |
| Wei et al. [19] | CPABE with secure user-revocation method | ✓ | X | X | X | X |
| Yu et al. [25] | Attribute based encryption policy with Multi-authority scheme | ✓ | X | X | ✓ | X |
| Cao et al. [22] | Permissionless blockchain with BLS-signature scheme | ✓ | ✓ | ✓ | ✓ | X |
| Huang et al. [26] | Permissioned blockchain with Public key infrastructure (PKI) | X | X | X | ✓ | X |
| Li et al. [27] | Permissioned blockchain with access authorization model | ✓ | X | X | ✓ | X |
| Li et al. [28] | Simple provenance assisted permissioned blockchain based storage | ✓ | X | X | ✓ | X |
| Hussein et al. [29] | Smart contract enabld ABE with IPFS | ✓ | X | X | ✓ | ✓(Due to IPFS based decentralised storage) |
| Dagher et al. [30] | Permissionless blockchain (PoW based) with HTTPs protocol | ✓ | X | X | ✓ | X |
| Benil et al. [31] | Permissioned blockchain with Public key infrastructure (PKI) | ✓ | X | X | ✓ | X |
| Yang et al. [48] | Consortium blockchain with secure attribute-based signature (ABS) | ✓ | X | X | ✓ | X |
| Sun et al. [49] | Permissioned blockchain blockchain with secure attribute-based encryption scheme | ✓ | X | X | ✓ | ✓(Due to IPFS based decentralised storage) |
| Chelladurai et al. [50] | Permissioned blockchain (PoW based) with MMHT scheme | ✓ | X | X | ✓ | X |
| Proposed | Permissionless blockchain (PoW based) with secure signature scheme | ✓ | ✓ | ✓ | ✓ | ✓(Due to IPFS based decentralised storage) |

has layered on the top of the Bit-torrent protocol and Kademlia Distributed Hash table (DHT) [36]. Moreover, Object Merkle DAG is perhaps the unique part of IPFS as the Content Addressing, Tamper Resistance, and Deduplication is all essential features from this distributed storage. IPFS also describes other artifacts to create a versioned file-system above the Merkle DAG [45]. IPFS considers that dividing large files into individual blocks is difficult, and IPFS provides many alternatives that the users can customize. In brief, IPFS is a secure, content-based addressing model for data storage mechanism, which provides high-capacity storage with efficient concurrent accessibility. In the proposed model, the EHRs are encrypted and stored in distributed IPFS nodes, whereas the corresponding hash-values are recorded in DHT and managed by the cloud manager (CM) at CSP.

## 4. Design Goals and Adversary model

In this section, we illustrate the system model, design goals and adversary model of the proposed methodology. Moreover, the notations used in this work are summarized in Table 2.

### 4.1. System model

As illustrated in Fig. 1, the system model consists of five elementary entities; the CSP, the doctors ($Dc_i$), the patients ($Pt_i$), the hospital $(Ht)$, and the trusted third party auditor (TPA). The CSP also consists of a sub-entity, which refers to cloud manager (CM). It offers the doctors both EHRs outsourcing and efficient decentralized EHR-storage. The patients refer to those who visit the hospital to meet the doctor for diagnosing their medical problems. The doctors refer to who generates the EHRs of the patients and outsources it to CSP. The hospital $(Ht)$ entity has all doctors and provides appointments to patients to meet with the doctors. On the other hand, a relevant entity named as TPA also exists in the model, which acts as a trusted authority between the CSP and other entities. Mainly, the TPA is described as the specialist in data-authentication. Thus, it can manage the auditing task to check the integrity of the outsourced EHRs. Initially, the patients register with the hospital and share some auxiliary information with the hospital. So, the hospital creates related diagnosing-information for the corresponding patient, which consists of information related to the appointed doctor with proper appoint-time, place, and some other necessary information. In contrast, the generated treatment_key (TK) is shared between the other entities. Afterwards, the patients meet with the doctor on the appointed time and diagnosed. Then, the doctors generate EHR for the patient and encrypt it; outsources it to CSP. Here, large-scale EHRs are stored in a decentralized manner at IPFS nodes in CSP. As a final step, the CSP authenticates the doctors by validating the correctness of the patient's meeting.

Further, the proposed model involves four different algorithms as follows;

**Table 2** Symbols table.

| Notations | Definitions |
|---|---|
| $l > 1$ | Security parameter |
| $p$ | Large prime number |
| $(G_1, G_2)$ | The multiplicative cyclic group of prime order $p$ |
| $g$ | generator of $G_1$ |
| $(ID_{Dc_i}, ID_{Pt_i})$ | Identities of doctors and patients, respectively |
| $(ID_{Ht_i})$ | Identity of the hospitals |
| $(\vartheta_i, P_{Dc_i})$ | Key-pair of doctors |
| $(\mu_i, Q_{Pt_i})$ | Key-pair of patients |
| $H, H_1, H_2$ | Collision resistant cryptographic function |
| $Pwd_{Pt_i}$ | Human-memorable password for patients |
| $Tm_{Pt}$ | Scheduled time of patients to meet appointed doctor |
| $Ct_i$ | Encrypted value of EHR $(ms)$ |
| $TK_i$ | Treatment key to take an appointment |
| $(WA_{Pt_i}, Wt_{Pt_i})$ | Patient's warrant to meet the doctor |
| $\sigma_i$ | Signature_value generated by doctor for $Ct_i$ |
| $\hat{Ct}_i$ | Outsourced value of patients' EHRs to cloud storage |
| $\Theta_i$ | Hash-value of outsourced EHRs by IPFS nodes |
| $(A_{CM}, A_{Dc})$ | Externally owned Ethereum accounts of CM and doctors |
| $Bl_{hash_{tm}}$ | The latest confirmed height of the blockchain |

1. Set_up procedure(): This randomized algorithm requires the security parameter ($l > 1$) as the input value, then generates system_parameter (*sys_param*), secret values, and initializes the whole system.
2. Appointment_procedure(): This randomized algorithm generates a treatment_key (TK) for each of the patients, then the patients get a scheduled appointment time from the hospital to meet the doctor.
3. Store_procedure(): This randomized algorithm facilitates the doctors to outsource the generated EHRs to CSP with its corresponding signature_value. Upon receiving the EHRs from doctors, the CSP store it over distributed IPFS nodes and integrates corresponding hash-values into a transaction on blockchain.
4. Auditing_procedure(): This randomized algorithm facilitates the auditor to verify the integrity of the outsourced EHRs. It also enables the CSP to demonstrate that the outsourced EHRs are effectively managed or not.

### 4.2. Adversary model

The adversary model can consider adversaries in two different perspectives; internal and external adversaries.

1. **Internal adversaries:** Generally, internal adversaries are related to semi-trusted doctors, rational CSP, and rational CM as follows:
   - **Rational CSP and CM:** In the proposed model, the CM is considered as a rational entity. As the CM is subject to CSP for the proposed model, this presumption adopts from the previously discussed cloud-based storage system [33–35]. Specifically, the CM can only deviate from the described model, iff its benefits will be increased. It may be incentivized by the malicious doctors to forge the outsourced EHRs, such as any malicious doctor may try to tamper the EHR to cover-up his/her medical malpractice.
   - **Semi-trusted doctors:** The doctors may be semi-trusted, as they would be trustworthy during the whole diagnosing period. However, after the completion of the diagnosing period, they would initiate severe attacks as follows:
     (I) The malicious doctor may break the privacy of outsourced EHRs. They can collude with malicious CSP to access the outsourced EHRs and try to reveal their sensible data or identity.
     (II) The malicious doctor may collude with any of the two entities, i.e., patients and the CSP, to outsources the forged EHR at CSP. Such as, after the completion of the diagnosing period, the malicious doctors may try to forge the outsourced EHR to cover-up their medical malpractices.

2. **External adversaries:** For external adversaries, the adversaries try to impersonate the patients for the appointment with the hospital. Notably, in the existing EHR storage system, the adversary often tries to request numerous diagnostic-token (i.e., tokens required to meet the doctor for diagnosing) and used these diagnostic-token for per-

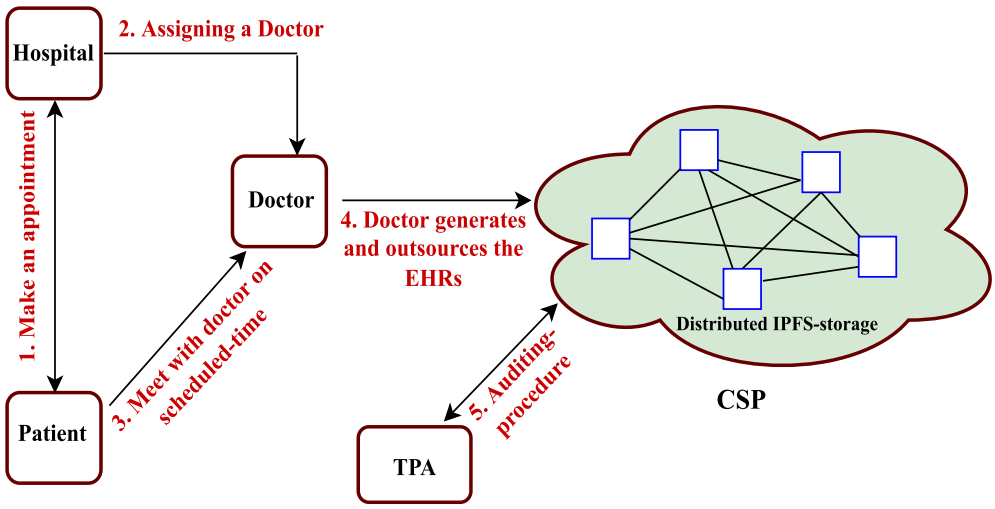


**Fig. 1** System model.

sonal benefits. At present, many cloud-based EHR storage systems implement a password-based authentication model to authenticate the patients and doctors. Such external adversaries may execute a password-guessing attack to break the infeasibility of such a cloud-based EHR storage system.

Moreover, for the sake of simplicity, the semi-trusted hospital is treated the same as the semi-trusted doctors in the adversary model of the proposed model.

### *4.3. Design goals*

To support an efficient tamper-proof blockchain-based approach for cloud-assisted EHR storage, the proposed model should attain the following properties;

1. **Accountability:** To prevent medical conflicts, the doctor needs to be accountable for the treatment he/she has made, and he/she can not modify or deny it. Anyone can check whether previous treatment reports have been modified.
2. **Efficiency:** The proposed model should be efficient with communication, computation, and storage overheads.
3. **Decentralized storage:** This feature assures that the outsourced EHRs are stored in a decentralized manner, which ensures the resistance towards unauthorized access and single-point of failure.
4. **Proper security:** The proposed model should satisfy the following security requirements:
   (I) Resistance against impersonation attack: Any external adversary may not impersonate any patient for making a scheduled appointment with the hospital by executing a password-guessing attack.
   (II) Resistance against any forgery attack: Any adversary cannot substitute the outsourced EHRs by forged ones for target ones (i.e., to hide any medical malpractice).
   (III) Resistance against any modification attack: Once the generated EHRs are properly outsourced and stored at CSP, any adversary cannot replace the existing EHRs, for target ones. Moreover, any malicious doctor can not incentivize the CSP to modify the outsourced EHR, which was generated by the same doctor, to hide his/her mistake.
   (IV) Confidentiality: Any adversary cannot be able to extract any part of outsourced EHRs.

## 5. Description of the proposed methodology

In the proposed model, initially, the patients make an appointment with the hospital and get the treatment_key (TK). Through this treatment_key (TK), a secure channel establishes between all the entities, i.e., hospital, patients, and the assigned doctors. This process follows the strong password-based authentication method with resistance to password-guessing attacks. So, it guarantees the computationally infeasibility of the proposed model towards the impersonate attacks by any external adversaries. Before the scheduled diagnosing time, the patients create a warrant to meet the doctor. This warrant consists of the details of the assigned doctors, i.e., doctor's ID ($ID_{Dc}$) with their diagnosing-time and other related information. During the diagnosing-time, the doctor properly examines the patient, generates the required EHR, and extracts the latest confirmed hash-values of $\varphi$ - successive blocks of the Ethereum blockchain [42]. After that, the doctor outsources the encrypted form of EHR with the corresponding signature to the distributed IPFS-nodes at CSP. The IPFS storage server automatically returns the hash-values of this outsourced EHR to the CM and store it to Distributed Hash Table (DHT) [36]. Then, the CM integrates this into transactions on Ethereum blockchain. Notably, in this framework, the hash-values of the outsourced EHRs are generated for the IPFS storage. With the help of these hash-values, the CM can easily detect any modifications to the outsourced EHRs in IPFS-nodes. Moreover, the doctors can be able to access the complete outsourced EHR in IPFS by the hash-values of the EHR on DHT. Further, these hash-values of outsourced EHRs are integrated into a transaction in Ethereum blockchain, which is generated within one diagnosing-time period for the patient. Despite the scenario as mentioned above, a different scenario may also exist where multiple patients - $Pt_i$, ($\forall_{i\in[1,\ldots,n]}$, where $n$-different

doctors and $n$-different patients) may visit multiple doctors ($Dc_i$) for treatment at the same time. All these doctors $Dc_i$ simultaneously generate a large number of EHRs for corresponding patients ($Pt_i$). Then, the doctors ($Dc_i$) outsources these EHRs with the corresponding signature to the distributed IPFS-node at CSP. Afterwards, IPFS storage generates hash-values for all these outsourced EHRs and integrates these hash-values into a transaction in Ethereum blockchain. Therefore, the proposed model supports batch outsourcing of generated EHRs, the whole procedure for batch outsourcing in Section 5.2. Furthermore, the timeliness of the EHRs also considers, as it is more significant to know when these EHRs were generated than what they were. The doctor entity integrates all the generated EHRs into a transaction on Ethereum blockchain. So, only by extracting the timestamp_value ($Tm$) from the particular block on Ethereum blockchain, which includes the corresponding transaction, anyone can get the time whenever the EHRs are generated. Moreover, to ensure the non-equivocation nature of the proposed model, we design the latest non-membership proof on the Ethereum blockchain. All the accounts created to generate the transactions in the proposed model are specially designed and committed, which facilitates the TPA to verify whether the number of generated transactions same as the number of EHRs recorded or not. It can be easily obtained from the "Nonce_value" of the account in Ethereum blockchain [51]. The security of the proposed model is specially designed on the computationally intractability of Ethereum blockchain. So, an adversary cannot fork the blockchain without having a significant percentage of the full computational power of the network. Even though the malicious CSP or CM collude with any malicious doctor to modify the outsourced EHRs, it cannot tamper the corresponding transactions on Ethereum blockchain. The complete illustration of the proposed model is described in Fig. 2.

### 5.1. Construction of the proposed model

The proposed methodology consists; a patient *(Pt)* with associated identity $ID_{Pt}$, a hospital *(Ht)* with associated identity $ID_{Ht}$, the doctor *(Dc)* with associated identity $ID_{Dc}$ and a TPA. The complete construction is divided into four phases; Set_up procedure (), Appointment_procedure (), Store_procedure() and Auditing_procedure(). The draft of these four phases is described in the following manner.

#### 5.1.1. Set_up procedure( )

For a given security parameter $(l > 1), G_1$ and $G_2$ be the two multiplicative cyclic group of prime order $p$ and g is the fixed generator of $G_1$. Let, the bilinear pairing as $e : G_1 \times G_1 \rightarrow G_2$. Further, selects three full-domain collision resistant secure random cryptographic hash functions, i.e., $H_1, H_2 : \{0,1\}^* \rightarrow G_1$, and $H : \{0,1\}^* \rightarrow Z_p^*$. Every doctor generates their personal externally owned account ($A_{Dc}$) in Ethereum blockchain and broadcasts it for others over the blockchain network. The CM also generate their personal externally owned account ($A_{CM}$) in Ethereum blockchain and send it to the TPA and the doctors. The hospital $Ht$ provides a human-memorisable-password $Pwd_{Pt}$ to the patients $Pt$. The patients $Pt$ randomly selects $\mu$as a patient's secret key, and also generates corresponding patient's public key as $Q_{Pt} = \mu \cdot g$. On the other hand, the doctor $Dc$ also randomly selects $\vartheta \in Z_p^*$ as a doctor's secret key and sets $P_{Dc} = \vartheta_i \cdot g$ as doctor's public key. Then, the system_parameter ($sys\_param$) is; $\{G_1, G_2, g, p, e, H, H_1, H_2, A_{CM}, A_{Dc}\}$.

#### 5.1.2. Appointment_procedure( )

In this phase, the patient $Pt$ gets the information related to appointment with the hospital $Ht$ under the protection of treatment_key (TK) as follows;

- Patient $Pt$ randomly selects $\alpha \in Z_p$, calculates $X = g^{\alpha}$ and $X_{Pt} = X(H_1(ID_{Pt}))^{Pwd_{Pt}}$. Afterward, the patient sends $X_{Pt}$ to hospital $Ht$.
- Hospital $Ht$ randomly selects $\beta \in Z_p$, calculates $Y = g^{\beta}$ and $Y_{Ht} = Y(H_1(ID_{Ht}))^{Pwd_{Pt}}$. Then, the hospital $Ht$ sends $Y_{Ht}$ to patient $Pt$.
- After receiving, $Y_{Ht}$ from the hospital $Ht$, patient $Pt$ computes, $K_{Pt} = \left(\frac{Y_{Ht}}{H_1(ID_{Ht})^{Pwd_{Pt}}}\right)^{\alpha}$ and generate treatment_key $(TK) = H(ID_{Pt}, ID_{Ht}, X_{Pt}, Y_{Ht}, Pwd_{Pt}, K_{Pt})$.
- The hospital $Ht$ computes $K_{Ht} = \left(\frac{X_{Ht}}{H_1(ID_{Ht})^{Pwd_{Pt}}}\right)^{\beta}$ and generates treatment_key as $(TK) = H(ID_{Pt}, ID_{Ht}, X_{Pt}, Y_{Ht}, Pwd_{Pt}, K_{Pt})$. This procedure helps the patient $Pt$ to make an appointment with the hospital $Ht$.

After, the hospital $Ht$ assigns a doctor $Dc$ to patient $Pt$, sends this appointment-information (under the protection of treatment_key (TK)) to patient $Pt$ and treatment_key (TK) to doctor $Dc$. Finally, the patient $Pt$ decrypts this appointment-information to obtain the information about the doctor $Dc$, the scheduled-time ($Tm_{Pt}$), and some other related-information ($Aux_{Pt}$).

#### 5.1.3. Store_procedure ( )

This phase is related to the storage of outsourced EHRs at CSP. Initially, the patient $Pt$ calculates a warrant ($Wt_{Pt}$) to meet the doctor $Dc$, where

$$WA_{Pt} = (ID_{Pt}||ID_{Dc}||Tm_{Pt}||Aux_{Pt}), \text{and} \quad Wt_{Pt} = \mu \cdot H_2(WA_{Pt}).$$

Then, the doctor $Dc$ generates the related EHR ($ms$) and encrypts it by a secure symmetric encryption technique i.e., AES-128 bit, and generates $Ct = E_k(TK, (ms||WA_{Pt}||Wt_{Pt}))$. Where $E_k$ represents a secure symmetric encryption technique i.e., AES-128 bit. Based on the current time $tm$, the doctor $Dc$ derives the hash-values of $\varphi$ - successive blocks which are latest confirmed on the Ethereum blockchain. These hash-values are described as $Bl_{hash_{tm}} = (Bl_{t-\varphi+1}, Bl_{t-\varphi+2}, \ldots, Bl_{t+\varphi})$, where $(t + \varphi)$ describes as latest confirmed height on the blockchain which are $\varphi \geqslant 12$ for Ethereum blockchain. Moreover, the doctor $Dc$ with associated identity ($ID_{Dc}$) computes

$$Q_m = H_1(ID_{Dc}), S = \vartheta \cdot Q_m, \text{ and}$$
$$\delta = H(Bl_{hash_{tm}}||H_1(ID_{Dc})||H_1(ID_{Pt})||H_2(Ct||Wt_{Pt})||WA_{pt}))$$

Then, the doctor $Dc$ randomly selects $r \in Z_p^*$, and computes

$$R = r \cdot g, \quad h = H_2(R, \delta, ID_{Dc}), \text{and} \quad V = r \cdot h + S$$

After that, the doctor $Dc$ calculate the signature_value as $\sigma = (V, R, \delta, ID_{Dc})$. Afterward, the doctor $Dc$ sends $\hat{Ct} = \{Ct, \sigma, Bl_{hash_{tm}}, WA_{Pt}, Wt_{Pt}\}$ to CSP. Upon receiving $\hat{Ct}$

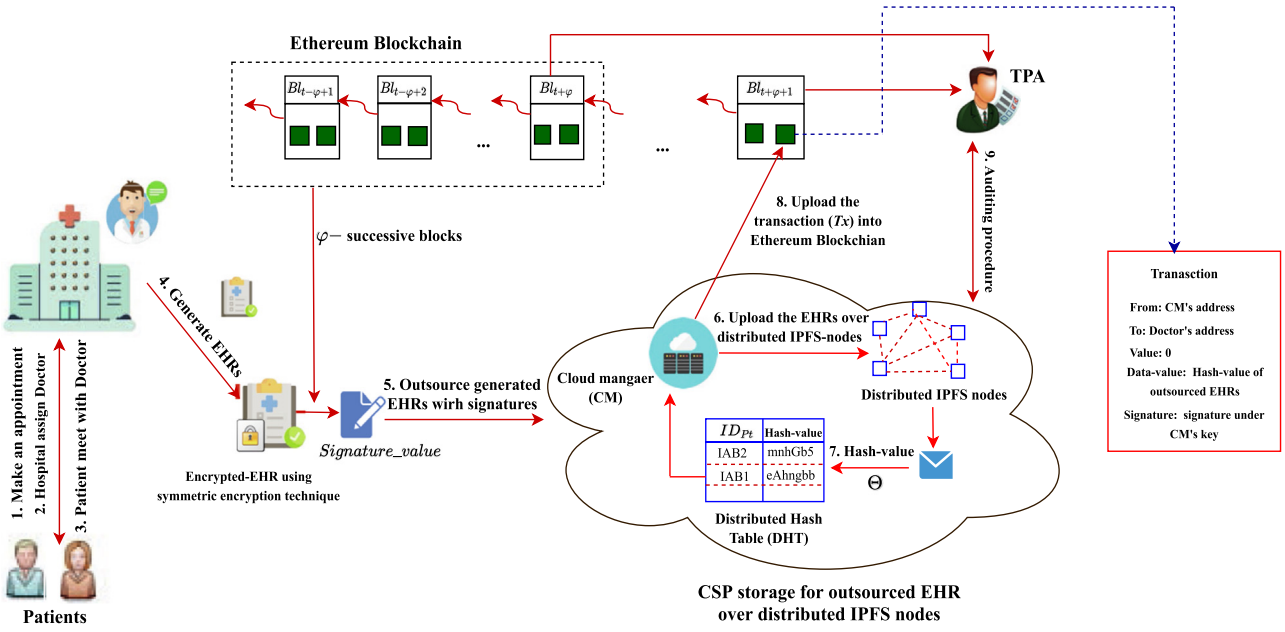


**Fig. 2** Illustration of complete EHR storage model.

from the doctor's side, the CM first confirms that $Bl_{t+\varphi}$ is the latest confirmed block on the Ethereum blockchain. If this confirmation fails, then the CM rejects it; otherwise, the CM verifies the following equation as follows;

$$e(V,g) \stackrel{?}{=} e(Q_m, P_{Dc}) \cdot e(R,h) \tag{1}$$

iff Eq. 1 holds, the CM accepts $\hat{Ct}$, otherwise rejects it.

Finally, the CM uploads this outsourced EHRs to IPFS storage on the cloud-environment in a decentralized manner. The IPFS storage store these EHRs into corresponding storage-node by using related information, i.e., patient-identity $(ID_{Pt})$ and automatically returns a hash-value $(\Theta)$, i.e., $\Theta = H_1(Ct||\sigma||Bl_{hash_{tm}})$, which is kept in DHT as described in Fig. 2.

Now, the CM creates the transactions $Tx_1$, as described in Fig. 3, where CM transfers 0 ether from their account $(A_{CM})$ to doctor's account $(A_{Dc})$, and $\Theta$ sets as data-value of transaction $Tx_1$. Afterwards, the CM uploads this transaction $Tx_1$ to Ethereum blockchain. This transaction $Tx_1$ is recorded as a latest block of height $(t+\varphi+1)$ [38,42]. Once the blockchain accepts and confirms the transaction, then the CM properly manages the outsourced EHRs at CSP over the distributed IPFS-storage node.

#### 5.1.4. *Auditing_procedure( )*

Given the EHR $(Bl_{hash_{tm}}, ID_{Pt}, Ct, WA_{Pt}, Wt_{Pt})$, the TPA can check the integrity of the outsourced EHR and timeliness of Ethereum blockchain as follows;

- Based on the stored $\hat{Ct}$ at the CSP, the TPA access the outsourced EHR. After that, the TPA parse the EHR and obtain $(Ct, WA_{Pt}, Wt_{Pt}, \sigma)$.
- TPA extracts the related transaction from Ethereum blockchain and obtain the information about the accounts of CM $(A_{CM})$ and account of doctor $(A_{Dc})$ from Ethereum blockchain. Further, based on the Nonce_value of $(A_{CM})$, the TPA also extracts the number of transactions performed from $(A_{CM})$ to $(A_{Dc})$.
- TPA verifies whether the number of generated transactions matches the number of EHRs recorded by the doctors. If the verification fails, the TPA rejects it.
- Afterward, TPA checks the validity of warrant $(Wt_{Pt})$ and $(\sigma)$; then, if this verification fails, the TPA rejects it.
- Validate the timeliness of the outsourced EHRs by verifying the transaction_time, which is obtained from the corresponding block on Ethereum blockchain, if the validation fails, the TPA rejects it.
- Based on the height of the block- $(t+\varphi+1)$, the TPA locates the block and derive the value of $\theta'$ from the corresponding transaction; then, compute $H_1(Ct||\sigma||Bl_{hash_{tm}}) = \theta'$ and verify $\theta \stackrel{?}{=} \theta'$. If the verification fails, the TPA rejects it.

### 5.2. *Support for batch outsourcing of EHRs*

In real scenario, multiple patients $(Pt_i)$ may visit to multiple doctors $(Dc_i)$, so multiple doctors generate and outsource many EHRs simultaneously. The individual outsourcing of these generated EHRs could be cumbersome, inefficient and unreliable for the CM and doctors. We assume that there are $n$-different patients $(Pt_1, Pt_2, \ldots, Pt_n)$ with associated identities $(ID_{Pt_1}, ID_{Pt_2}, \ldots, ID_{Pt_n})$ and $n$-different doctors $(Dc_1, Dc_2, \ldots, Dc_n)$ with associated identities $(ID_{Dc_1}, ID_{Dc_2}, \ldots, ID_{Dc_n})$. Multiple patients may get appointment for multiple doctors simultaneously. Afterwards, the particular patient $Pt_i$ for $(i \in n)$, meet to their assigned doctor $(Dc_i)$ on the scheduled-time. Then, the corresponding doctor generates the EHRs and outsources it. However, it is more favorable for the CM to batch all these outsourcing tasks together and store all outsourced EHRs over the distributed IPFS nodes at CSP

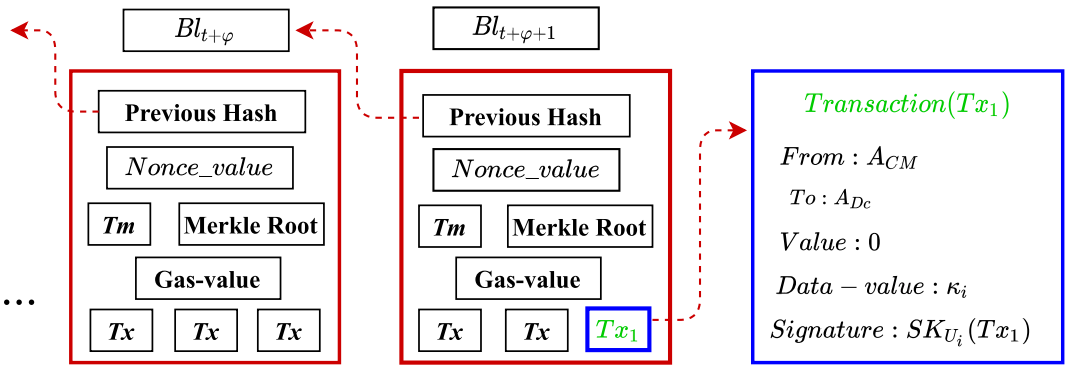


**Fig. 3** Transaction $(Tx_1)$ for outsourcing of single EHR.

at one-time. Therefore, we enhance the proposed model to support batch outsourcing and auditing, which is illustrated in the following manner.

*5.2.1. Set_up procedure( )*

As described earlier, the hospital $Ht$ provides a human-memorisable password - $Pwd_{Pt_i}$ $\forall_{i\in[1,\ldots,n]}$ for every patients $(Pt_i)$. The patients $(Pt_i)$ randomly selects $\mu_i$as its secret key, and also generates corresponding patient's public key as $Q_{Pt_i} = \mu_i \cdot g$. On the other hand, the doctor $Dc_i$ also randomly selects $\vartheta_i \in Z_p^*$ as its secret key and sets $P_{Dc_i} = \vartheta_i \cdot g$ as its public key, $\{A_{CM\_sender}, A_{CM\_receiver}\}$ - are two accounts of CM on Ethereum blockchain, and the remaining parameters are similar to those mentioned above. Thus, the system_parameter $(sys\_param)$ is $\{G_1, G_2, g, p, e, H, H_1, H_2, A_{CM\_sender}, A_{CM\_receiver}\}$.

*5.2.2. Appointment procedure( )*

The patients $(Pt_i)$ get the appointment with hospital $(Ht)$ under the protection of corresponding treatment_key (TK) as follows;

- Each of the patient $Pt_i$ randomly selects $\alpha_i \in Z_p$, calculates $X_i = g^{\alpha_i}$ and $X_{Pt_i} = X_i(H_1(ID_{Pt_i}))^{Pwd_{Pt_i}}$. After that, the patient sends $X_{Pt_i}$ to hospital $Ht$.
- Hospital $Ht$ randomly selects $\beta \in Z_p$, calculates $Y = g^{\beta}$ and $Y_{Ht_i} = Y(H_1(ID_{Ht}))^{Pwd_{Pt_i}}$. After that, the hospital $Ht$ sends $Y_{Ht_i}$ to patient $Pt_i$.
- After receiving, $Y_{Ht_i}$ from the hospital $Ht$, every patient $Pt_i$ computes, $K_{Pt_i} = \left(\frac{Y_{Ht_i}}{H_1(ID_{Ht})^{Pwd_{Pt_i}}}\right)^{\alpha}$ and generate their corresponding treatment_key $(TK_i) = H(ID_{Pt_i}, ID_{Ht}, X_{Pt_i}, Y_{Ht_i}, Pwd_{Pt_i}, K_{Pt_i})$.
- The hospital $Ht$ computes, $K_{Ht_i} = \left(\frac{X_{Ht_i}}{H_1(ID_{Ht})^{Pwd_{Pt_i}}}\right)^{\beta}$ and generate treatment_key $(TK_i) = H(ID_{Pt_i}, ID_{Ht}, X_{Pt_i}, Y_{Ht_i}, Pwd_{Pt_i}, K_{Pt_i})$, and patient $Pt$ makes an appointment with the hospital $Ht$.

After, the hospital $Ht$ assigns a doctor $(Dc_i)$ to particular patient $(Pt_i)$, sends this appointment-information (under the protection of corresponding treatment_key $(TK_i)$ to patient $(Pt_i)$ and treatment_key $(TK_i)$ to doctor $(Dc_i)$. Finally, the each of the patient $(Pt_i)$ decrypts this appointment-information to obtain the information about their respective doctor $(Dc_i)$, the scheduled-time $(Tm_{Pt_i})$, and some other related-information $(Aux_{Pt_i})$.

*5.2.3. Store_procedure ( )*

As described earlier, each of the patient $(Pt_i)$ calculate their respective warrants $(Wt_{Pt_i})$ to meet their respective assigned doctor $(Dc_i)$, i.e.,

$$WA_{Pt_i} = (ID_{Pt_i}||ID_{Dc_i}||Tm_{Pt_i}||Aux_{Pt_i}), Wt_{Pt_i} = \mu_i \cdot H_2(WA_{Pt_i}).$$

Then, the respective assigned doctor $(Dc_i)$ for each of the patients $(Pt_i)$ generates the related EHRs $(ms_i)$ and encrypts it by a secure symmetric encryption technique, i.e., AES-128 bit, and generates $Ct_i = E_k(TK_i, (ms_i||WA_{Pt_i}||Wt_{Pt_i}))$ Based on the current time $tm$, the doctor $(Dc_i)$ derives the hash-values of $\varphi$ - successive blocks which are latest confirmed on the Ethereum blockchain. This hash-values is described as $Bl_{hash_{tm}} = (Bl_{t-\varphi+1}, Bl_{t-\varphi+2}, \ldots, Bl_{t+\varphi})$. Moreover, the doctor $(Dc_i)$ with associated identity $(ID_{Dc_i})$ computes

$$Q_{m_i} = H_1(ID_{Dc_i}), S_i = \vartheta_i \cdot Q_{m_i}, \text{ and}$$
$$\delta = H(Bl_{hash_{tm}}||H_1(ID_{Dc_i})||H_1(ID_{Pt_i})||H_2(Ct_i||Wt_{Pt_i})||WA_{pt_i}))$$

After that, the doctor $(Dc_i)$ randomly selects $r_i \in Z_p^*$, and computes

$$R_i = r_i \cdot g, h_i = H_2(R, \delta, ID_{Dc_i}), V_i = r_i \cdot h_i + S_i$$

Then, the doctor $(Dc_i)$ calculates the signature_value as $\sigma_i = (V_i, R_i, \delta, ID_{Dc_i})$. Afterward, the doctor $(Dc_i)$ sends $\hat{Ct}_i = \{Ct_i, \sigma_i, Bl_{hash_{tm}}, WA_{Pt_i}, Wt_{Pt_i}\}$ to CSP. Upon receiving $\hat{Ct}_i$ from the doctor's side, the CM first confirms that $Bl_{t+\varphi}$ is the latest confirmed block on the Ethereum blockchain. In the batch, outsourcing of EHRs from multiple doctors who select the same $Bl_{hash_{tm}}$ are outsourced simultaneously. If this confirmation fails, then the CM rejects it; otherwise, the CM verifies the following equation as follows;

$$e(\sum_{i=1}^{n} V_i, g) \stackrel{?}{=} \prod_{i=1}^{n} e(Q_{m_i}, P_{Dc_i}) \cdot e(R_i, h_i) \tag{2}$$

iff Eq. 2 holds, then the CM accepts $\hat{Ct}_i$, otherwise rejects it. Finally, the CM uploads this outsourced EHRs to IPFS storage on the cloud-environment in a decentralized manner. The IPFS will store this EHRs into corresponding storage-node by using related information, i.e., patient-identity $(ID_{Pt_i})$ and automatically returns a hash-value $(\Theta_i)$, i.e., $\Theta_i = H_1(Ct_i||\sigma_i||Bl_{hash_{tm}})$.

Now, the CM creates the transactions-$Tx_1$ as described in Fig. 4. Where CM transfer 0 ether from $(A_{CM\_sender})$ to $(A_{CM\_receiver})$, and $\Theta$ sets as data-value of transaction-$Tx_1$.

Afterward, the CM uploads this transaction-$Tx_1$ to Ethereum blockchain. This transaction-$Tx_1$ is recorded as a latest block of height $(t+\varphi+1)$. Once the blockchain accepts and confirms the transaction, the CM appropriately manages the outsourced EHRs at CSP over the distributed IPFS-storage node.

*5.2.4. Auditing_procedure ( )*

This phase is same as "single outsourcing of EHR for a single patient by a single doctor". So the whole auditing_procedure would not repeat for the sake of simplicity.

## 6. Security analysis

In this section, a brief explanation about the security analysis of the proposed model is evaluated with detailed proofs of various theorems.

**Theorem 1.** The proposed model is computationally intractable against impersonation attacks.

**Proof.** In the proposed model, impersonation attacks executed by any external adversary can be considered when the adversary $A$ may obtain the treatment_key (TK) executes password-guessing attacks. Whenever the adversary $A$ is successful, then he/she may impersonate the target for making the appointment. As a result, the adversary $A$ may execute various attacks, i.e., Distributed Denial-of-Service (DDoS), to tamper the proposed model. We describe the security analysis of the password-based authenticated key-exchange protocol as follows. To show the computational infeasibility of password-based authenticated key-exchange protocol, we follow the multi-party key-exchange protocol model described in [39,40]. In this model, the simulator will try to find guesses on the password by analyzing the random oracle queries of the adversary $A$. Afterwards, this information can be used to break the computational intractable strong Decisional Diffie-Hellman (DDH) - assumption [41]. Thus, we can prove that if the adversary $A$ can execute the prohibited queries with non-negligible probability. Then, the adversary $A$ can break the computational intractable strong DDH-assumption with non-negligible probability.

Suppose, the misleading query is performed under $W$-queries, where $W$ be a polynomial and limited by the execution-time of the adversary $A$. Let $v_1 = g^a, v_2 = g^b$, and $v_3 = g^c = G_p$ be a strong DDH-assumption. The simulator randomly selects $i \in [0, W]$. For the $i^{th}$- delivery query-message, to start a session between the patient and the hospital, the simulator sets $X_{Pt} = v_1 \cdot (H_1(ID_{Pt}))^{Pwd_{Pt}}$. It indicates that the patient $Pt$ randomly selects $x$ to replace the value of $a$, whereas the simulator doesn't have any idea for $a$. Whenever, $Ht$ make a respond to $X_{Pt} = x \cdot (H_1(ID_{Pt}))^{Pwd_{Pt}}$. Then, $Ht$ set $Y_{Ht} = v_2 \cdot (H_1(ID_{Ht}))^{Pwd_{Pt}}$. It indicates that the hospital $Ht$ randomly selects $y$ to substitute the value of $b$. At this situation, if the adversary $A$ generates a set of queries to random oracle $H_1$ with

$$X_{Pt} = x \cdot (H_1(ID_{Pt}))^{Pwd_{Pt}}, Y_{Ht} = v_2 \cdot (H_1(ID_{Ht}))^{Pwd_{Pt}}$$

and treatment_key (TK) = $v_3$, then guess that the DDH-assumption is a valid Diffie-Hellman (DH) instance such that $v_3 = g^{ab}$. Otherwise, the simulator flips a coin to conclude either $v_3 = g^{ab}$ or not. Let $\in$ be the adversary's probability for breaking the real instances. As the probability of $v_3 = g^{ab}$ is the same as to randomly select $v_3$ in the defined DDH-assumption, so the adversary can break the computationally intractable strong DDH-assumption with the non-negligible probability -

$$\left[\frac{1}{2} + \frac{\in}{4W}\left(\frac{1}{2}\left(\frac{1}{2}\left(1 - \frac{\in}{W}\right) + \frac{\in}{W}\right) + \frac{1}{2}\left(\frac{1}{2}\right)\right)\right]$$

where $\left(\frac{1}{2} + \frac{\in}{4W}\right)$ represents half probability of the execution of random oracle queries by the external adversary for $(Pwd_{Pt_i})$ which includes $v_3$ in random oracle and 4 in the denominator of $\frac{\in}{4W}$ comes from the half probability of DDH-assumption which is true for the defined DDH-instance. Further, the tuple $\left(\frac{1}{2}\left(\frac{1}{2}(1 - \frac{\in}{W}) + \frac{\in}{W}\right)\right.$ represents the probability of the external adversary to satisfies $(g^a, g^b, g^{ab})$ for randomly selected $g^c$ with random coin-flips and the value $\frac{1}{2}\left(\frac{1}{2}\right)$ shows the total probability of the external adversary's success for $v_3 = g^{ab}$.

**Theorem 2.** The proposed model is computationally secure against any type of modification attack on outsourced EHRs at CSP.

**Proof.** For the proposed model, we employ three security techniques to resist any type of modification attacks for outsourced data as follows;

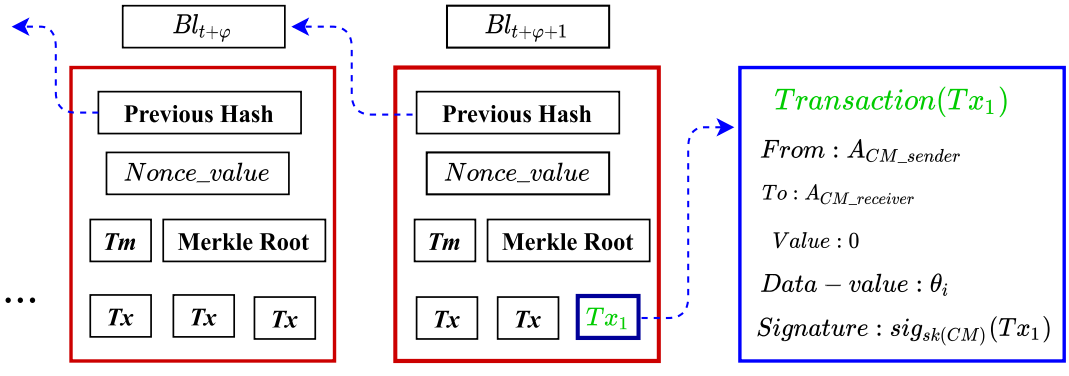


**Fig. 4** Transaction $(Tx_1)$ for batch outsourcing of multiple EHRs.

1. **First technique:** This is related to computationally infeasibility of the signature_value against any modification. The doctors ($Dc_i$) generates the signature_value ($\sigma_i$) for all the generated EHRs and outsource it with EHRs to CSP. Then, none of the probabilistic polynomial-time (PPT) adversary $A$would be able to modify the signature_value ($\sigma_i$).

The proper computationally infeasibility of signature_value ($\sigma_i$) against modification attack is formed by simulating a game (*Game* 1) between a probabilistic polynomial-time (PPT) adversary $A$ and TPA. Now, we can prove that if the adversary $A$ is able to perform any modification with non-negligible success probability on ($\sigma_i$), then the algorithm $C$ can able to generate solution of Computational Diffie–Hellman (CDH)-problem in $G_1$ [41,44].

*Game* 1: $A$ can obtain following three oracle queries as;

1. Reveal_sys_param_Query - $A$ can request for sys_param from TPA. So, the TPA returns sys_param as a response to $A$.
2. Random_oracle_queries - $A$ can request random_oracle_-queries ($H_i|i = 1,2$) any-time. Then, $A$ maintain a two lists - ($H_1^{list}$ and $H_2^{list}$) to respond to these queries.
3. Signing_queries - $A$ can request signing_queries for ($ID_{Dc_i}, \delta_i$). The TPA responds to these queries with ($\sigma_i$).

Further, *Game* 1 has three different phases as follows;

1. Phase 1: Initially, $A$ creates a reveal_sys_param_query. Afterward, the TPA sends sys_param to $A$ as a reponse.
2. Phase 2: In this phase, $A$ can make random_oracle_query and signing_queries.
3. Phase 3: Finally, after a significant number of random_oracle_queries, $A$ is capable to create $\sigma_i^* = (V_i^*, R_i^*, ID_{Dc_i}^*, \delta_i^*)$.

$A$ wins the *Game* 1 iff;

(a) $\sigma_i^*$ is a valid signature_value on ($ID_{Dc_i}, \delta_i$).
(b) $A$ has not issued any signing_queries for ($ID_{Dc_i}, \delta_i$).

The security of $\sigma_i$ is defined as follows;

**Definition 4:** Any probabilistic polynomial-time (PPT) adversary$A$ $(t, l_i, l_s, \in)$ - breaks the signature_value ($\sigma_i$) if $A$ execute at most - t times, creates at most $l_s$ - signing_queries, at most $l_i$ - random_oracle_queries to the hash functions, and $Adv_{sig_A}$ is at least $\in$. Thus, the signature_value ($\sigma_i$) is $(t, l_i, l_s, \in)$ - existentially unforgeable under the adaptive chosen-message attack if none of the probabilistic polynomial-time adversary $A$ $(t, l_i, l_s, \in)$ - breaks the above defined *Game* 1 with negligible probability. The following lemma shows that the signature_-value ($\sigma_i$) is computationally infeasible.

**Lemma 1:** Let $H_1$ and $H_2$ be two random oracles and also exist an adversary $A$ against the signature_value ($\sigma$) with $\in$-advantage while executing in $t$- time. Then, there exists an algorithm $C$ to solve the strong CDH-problem with probability

$$\in'' \geqslant \left(\frac{1}{(1+l_s)e}\right) \cdot \in$$

and executing in time ($t''$)

$$t'' \leqslant t + (l_1 + l_2 + 4l_s) \cdot T_{sm}$$

describes as time required for calculating the scalar-multiplication, $l_i$ times random_oracle_queries to $H_i$ ($i$ = 1,2), and $l_s$ as number of signing_queries by the doctor.

**Proof:** Initially, we assume that the adversary $A$ is capable to break the strong EUF-CMA (i.e., existentially unforgeable against adaptive chosen message attacks [43]) security of the signature-scheme. Then, we design an algorithm $C$ with a random instance $(g, x \cdot g, y \cdot g)$ to solve the strong CDH-problem [41]. Further, we can prove that how the algorithm $C$ can use the adversary $A$ to get the value of $(x \cdot y \cdot g)$ in $G_1$.

A. **Reveal_sys_param_Query:**Firstly, the algorithm $C$ sets $Q_{Pt} = x \cdot g$, and generates the system_parameter (*sys_param*) as $\{G_1, G_2, g, p, e, H_1, H_2\}$. The algorithm $C$ controls the hash-functions, i.e., $H_1$ and $H_2$, which are operated as random oracles. The algorithm $C$ randomly selects a integer-index $\alpha \in [1, l_1]$, where the $\alpha^{th}$-query to $H_1$ is on the specified $ID_{Dc}^*$. The algorithm $C$ replies to the adversary $A$'s queries as follows (i.e., these queries involve $H_i$ ($i$ = 1,2) queries and signing_-queries, both queries/replies pairs are maintained in lists.).

B. **Random_oracle_queries ($H_i$ ($i$ = 1,2) queries)** Upon receiving a query on $ID_{Dc_i}$, $C$ maintains a list $H_1^{list}$ of tuples $(h_{1i}, ID_{Dc_i}, Q_{m_i})$ and executes the following steps;

- if $i \neq \alpha$, $C$ randomly selects $h_{1i} \in Z_p^*$ and calculates $Q_{m_i} = H_1(ID_{Dc_i}) = h_{1i} \cdot g$, i.e., the probability for $i = \alpha$is $(P_{i=\alpha} = \xi)$ and the probability for $i \neq \alpha$is $(P_{i\neq\alpha} = 1 - \xi)$;
- else if $i = \alpha$, set $h_{1\alpha} = \perp, Q_{m_\alpha} = y \cdot g$;
- then, $C$ maintains the corresponding tuples to $H_1^{list}$.

We also assume that the adversary $A$ often executes the associated $H_1$ - queries, before executing the other related queries. Further, upon receiving the $H_2$-query on tuples $(R_i, ID_{Dc_i}, \delta_i)$, $C$ maintains a list $H_2^{list}$ of tuples $(R_i, ID_{Dc_i}, h_{2i}, h_{2i} \cdot g, \delta_i)$, randomly selects $h_{2i} \in Z_p^*$, and calculates $H_2(R_i, ID_{Dc_i}, \delta_i) = h_{2i} \cdot g$. Then, $C$ maintains the corresponding tuples to $H_2^{list}$. Whenever, the doctor $Dc$ receives the query on an identity- $ID_{Dc_i}$, then generates;

- if $i = \alpha$, $C$ aborts the game;
- else, $C$ executes $H_1$ - queries on $ID_{Dc_i}$, then checks $H_1^{list}$ for a tuple $(h_{1i}, ID_{Dc_i}, Q_{m_i})$, and computes $S_i = h_{1i} \cdot g$.

C. **Signing_queries:** Upon receiving the signing_queries on $(ID_{Dc_i}, \delta_i)$ is as follows;

- if $i \neq \alpha$, $C$ normally executes the signing procedure to generate the $\sigma_i$.
- if, $i = \alpha$, then $C$ generates the $\sigma_i$;
  - (a) $C$ randomly selects $r_1, r_2 \in Z_p^*$;
  - (b) Calculate $R_i = r_1 \cdot g, V_i = r_2 \cdot g$;
  - (c) Set $H_2(R_i, ID_{Dc_i}^*, \sigma_i) = r_1^{-1}(r_2 \cdot g - H_1(ID_{Dc_i}^*))$.If $H_2^{list}$ has tuple $(R_i, ID_{Dc_i}^*, \perp, \delta_i)$, then $C$ selects a another $r_1 \in Z_p^*$ and repeatedly execute the above mentioned steps.
  - (d) The signature_value is $(V_i, R_i, \delta_i, ID_{Dc_i})$.

D. **Forgery performed by adversary** $A$**:** Finally, the adversary $A$ performs an attack by forging the signature-value, i.e., $\sigma_i^* = (V_i^*, R_i^*, ID_{Dc_i}^*, \delta_i^*)$. If the adversary $A$ successfully forges the signature_value $\sigma_i^*$, then $\sigma_i^*$ is valid and holds the verification-condition within polynomial-time;

$$e(\sum_{i=1}^{n} V_i^*, g) = \prod_{i=1}^{n} e(Q_{m_i}^*, v_i^* \cdot g) \cdot e(R_i^*, h_2^*),$$

where $Q_{m_i}^* = H_1(ID_{Dc_i}^*) = y \cdot g, h_2^* = H_2(ID_{Dc_i}^*,$ and $\delta_i^*, R_i^*)$. Then, check the $H_2^{list}$ for $H_2(ID_{Dc_i}^*, \delta_i^*, R_i^*) = h_{2\alpha_i} \cdot g$, and $C$ can convert the above verification-condition;

$$e(\sum_{i=1}^{n} V_i^*, g) = \prod_{i=1}^{n} e(x \cdot g, y \cdot g) \cdot e(R_i^*, h_{2\alpha_i} \cdot g)$$

Thus, it is simple for $C$ to attain the solution of strong CDH-problem

$$x \cdot y \cdot g = V^* - h_{2\alpha_i} \cdot R_i^*.$$

Clearly, it is in contradiction with the intractability to solve the strong CDH-problem in polynomial time $t$. Moreover, to complete the proof, we can prove that $C$ solves the given CDH-problem with probability $\in'' \geqslant \left(\frac{1}{(1+l_s)e}\right) \cdot \in$. Initially, $C$ needs three different events to succeed:*E1*: Any probabilistic polynomial-time adversary$A$ aborts $C$ for any signing_query. *E2*: $A$ creates a valid and non-trivial signature_value $(\sigma_i^*)$ for $(ID_{Dc_i}, \delta_i)$.*E3*: $A$ creates valid and non-trivial modification and $C$ does not abort is probable.

$C$ succeeds iff all the above mentioned events occur. So,

$$P(E1 \cap E2 \cap E3) = P(E1) \cdot P(E2|E1) \cdot P(E3|E1 \cap E2)$$

where, $P(E1)$ represents the probability that for any signing_ queries $A$ does not abort $C$ is at least $(1-\xi)^{l_s}$,

$P(E2|E1)$ represents the probability that $C$ does not abort for any signing_queries and can produce a valid signature_-value for $(ID_{Dc_i}, \delta_i)$ is at least $\in$, and.

$P(E3|E1 \cap E2)$ represents the probability that any $A$ produces a valid modification for given *E1* & *E2* and $C$ does not abort is at least $\xi$.

**Proof:** Now, $P(E1 \cap E2 \cap E3) = (1-\xi)^{ls} \cdot \in \cdot \xi$. So,

$$\in'' = (1-\xi)^{l_s} \cdot \in \cdot \xi \tag{3}$$

$\in'_{min}$ can be obtained by differentiating Eq. 3

$$\frac{d\in''}{d\xi} = \frac{d((1-\xi)^{l_s} \cdot \in \cdot \xi)}{d\xi} = (1-\xi)^{l_s-1} \cdot \in \cdot (1-\xi(1+l_s))$$

Then, put $\frac{d\in''}{d\xi} = 0$, and we obtain $\xi_{opt} = 1/(1+l_s)$. So, Eq. 3 becomes $\in'' \geqslant \xi \cdot (1-\xi^{l_s}) \cdot \in \geqslant 1/(1+l_s)[1-(1/(1+l_s))] \cdot \in$.

For,

$$\lim_{l_s \to \infty} \left[1 - \frac{1}{(1+l_s)}\right] \to e$$

Thus, $P(E1 \cap E2 \cap E3)$ becomes $\in'' \geqslant \left(\frac{1}{(1+l_s)e}\right) \cdot \in$.

Moreover, the following proof illustrates that $C$ solve the computationally hard CDH-problem in polynomial time $t''$, i.e., $t'' \leqslant t + (l_1 + l_2 + 4l_s) \cdot T_{sm}$. Therefore, none of the probabilistic polynomial-time adversary$A$ can be able to modify the signature_value with non-negligible probability in polynomial time $t$.

**2. Second technique:** This is a secure assigning technique (i.e., assigning a doctor to a patient during the appointment procedure), where the patient $(Pt)$ generates a warrant $(Wt_{Pt})$ to meet the doctor. The generation of warrant $(Wt_{Pt})$ follows the security of computationally hard CDH-assumption. Then, none of the probabilistic polynomial-time adversary $A$ would be able to modify the warrant $(Wt_{Pt})$.

**Definition 5:** The $(t, \in)$ - CDH assumption holds in $G_1$ if no probabilistic polynomial-time adversary $A$, i.e., $t$ - time adversary $A$ has advantage $Advt_{G_1}^{CDH}$ at least - $\in$ in solving the CDH assumption in $G_1$ [41].

**Lemma 2:** Let $(G_1, G_2)$ be $(t'', \in'')$ - Co-CDH group pair of order p. Then, the warrant $(Wt_{Pt})$ on $(G_1, G_2)$ is $(t, l_s, \in)$ - secure against existential modification under an adaptive chosen-message attack, $\forall_{t,\in}$ satisfying $\in \geqslant e(l_s+1) \cdot \in''$ and $t \geqslant t'' - d_{G_1}(2d_s)$, where $d_{G_1}$ - represents a constant value $\in G_1$.

**Proof:** By using this lemma, we show that the counterfeiting of the warrant $(Wt_{Pt})$ is computationally hard. As described in the security analysis of [41,44], there is no probabilistic polynomial-time adversary $A$ has advantage $Advt_{G_1}^{CDH} \geqslant \in$, for solving the CDH assumption in polynomial time - $t$. So, the warrant $(Wt_{Pt})$ is computationally unforgeable until the CDH assumption is computationally hard in $G_1$. The detailed proof can be found in [41,44].

**3. Third technique:** This security technique is related to blockchain based tamper-resist technique, where the out-sourced EHRs, corresponding signature_value, and warrant are integrated into a transaction on a block in Ethereum blockchain. By the security definition of blockchain [38], none of the probabilistic polynomial-time adversary $A$ would be able to break the computational intractability of Ethereum blockchain.

**Lemma 3:** In a Proof-of-Work (PoW), based Ethereum blockchain, any probabilistic polynomial-time adversary $A$ cannot be able to modify the hash value of transaction $(Tx_1)$ in blockchain in polynomial time - $t$.

**Proof:** initially, we describe the pre-image resistance of any cryptographic hash-function, i.e., "if any given hash-function $H(\cdot)$ with output $Z$, it is computationally infeasible for any probabilistic polynomial-time adversary $A$ to find the value $y$, such that, $Z = H(y)$ [37]".

Therefore, by the pre-image resistant property of hash-function $H(\cdot)$. The proof of this lemma is straightforward.

Assuming, the latest block of height $(t+\varphi+1)$ of Ethereum blockchian in Fig. 3 and Fig. 4, where the data-value of $Bl_{t+\varphi+1}$ is $H_1(Ct||\sigma||Bl_{hash_{tm}})$ and $H_1(\cdot)$ is computationally secure hash-function with pre-image resistant. Therefore, if any probabilistic polynomial-time adversary $A$ can be able to modify the $H_1(Ct||\sigma||Bl_{hash_{tm}})$ of block - $(t+\varphi+1)$, the pre-image resistant of $H_1(\cdot)$ can be broken.

**Theorem 3.** The proposed model is computationally secure against any type of forgery attack on outsourced EHRs.

**Proof.** In the proposed model, any type of forgery attack can be categorized into two different types as follows;

• **Type 1. Resistance against the malicious CM**: In the case of malicious CM, we demonstrate that the proposed model is fully secure against equivocation attacks to assure the non-equivocation nature. In the proposed model, after the EHR is outsourced at CSP, the malicious CM can equivocate about the EHR on the blockchain to replace the existing transaction ($Tx_1$) with recently generated transaction ($Tx_1'$). The complete illustration of such attack is described in Fig. 5, where the CM generates a latest transaction ($Tx_1'$) and integrates the target EHR into this transaction ($Tx_1'$). Therefore, these two transactions (i.e., one transaction $Tx_1$, which is generated by doctor is for actual EHR, and another transaction ($Tx_1'$) is generated by the malicious CM for target EHR) on the blockchain relate to the same EHR can induce an equivocation. However, in the proposed model, such type of attack can be easily identified. Note that, whenever the TPA verifies the integrity of outsourced EHRs, firstly, it verifies whether the number of transactions generated by the CM is the same as the number of EHRs recorded, which have been stored in distributed IPFS nodes at CSP or not. If this verification fails, the integrity of the outsourced EHRs would be violated. As the $A_{CM}$ is specially designed and committed for generating the transaction, the number of transactions generated by the CM can easily be obtained from the "Nonce_value" of the block in $A_{CM}$. Therefore, the proposed model is fully resisting against malicious CM.

• **Type 2. Resistance against semi-trusted doctors:** In the proposed model, any semi-trusted doctor cannot perform forgery-attack on any outsourced EHR, even if the EHR is generated by the doctor herself/ himself. However, any semi-trusted doctor tries to forge any outsourced EHR, which may create two different cases of forge-attack as follows;

• **Case 1.** The first case considers as any malicious doctor colludes with the CM to substitute the existing EHRs with the latest ones.

**Proof:** As described earlier, each of the generated EHRs is integrated into a transaction ($Tx_1$) in the blockchain. So, whenever any malicious doctor wants to substitute the existing EHR with the latest one by himself/herself, then the only option for the doctors is to hard-fork the Ethereum blockchain. Specifically, for an instance of this attack, the Ethereum blockchain has the structure described in Fig. 6 by the red-blocks. The malicious doctor and CM generates the latest transaction - ($Tx_1'$) and tries to integrate $Tx_1'$ into chained blocks on the blockchain, which is illustrated with green blocks in Fig. 6. As every block in the Ethereum blockchain has *Previous Hash* (the hash-value of the previous block), if any malicious doctor and CM substitutes the target block with the latest generated block, it could break the Ethereum blockchain's validity. Therefore, there is only one way to perform this type of attack is to hard-fork the blockchain, i.e., the malicious doctor makes the latest blockchain with newly generated transaction ($Tx_1'$) on the block (presented inside the blue rectangle) as latest confirmed longest-chain. However, due to computationally intractability of Ethereum blockchain, it is computational-hard to fork the Ethereum blockchain by any adversary $A$ with limited power and hashrate $< 51\%$ of the overall network's hash rate. The concrete proof for the chain-growth property (i.e., longest chain property) of the blockchain follows the Lemma 4.

**Lemma 4:** In the proposed model, no probabilistic polynomial-time adversary ($A_I$) or malicious doctor can perform hard-fork over the blockchain with limited power.

**Proof:** Initially, we consider an assumption that the malicious forks can still happen even if there are no inadvertent forks. The malicious doctors or adversaries can secretly maintain a longer chain than the actual blockchain. Once it is released, it will immediately replace the real blockchain with the latest longest chain. To prove the validity of Ethereum blockchain against hard-fork, we describe a game. This game considers a set of honest doctors with CM and a set of malicious doctors with CM, respectively. For this game, the goal of the malicious doctor is to bias the probability distribution of the extract for the next block in the blockchain, which refers to as "Successive block ($\varphi = 12$)". Assume that the extract $m$ of the total successive block is a random pick. The malicious doctors win if $m$ belongs with a specific subset ($\iota$) of the winning pick. Let the characteristic function of such a subset of the set of feasible extract blocks are $\chi : \iota \rightarrow \{0, 1\}$. The malicious doctor wins the game if $\chi(m) = 1$. The probability of malicious doctor or adversary's winning is represented by $P_{A_I}$.

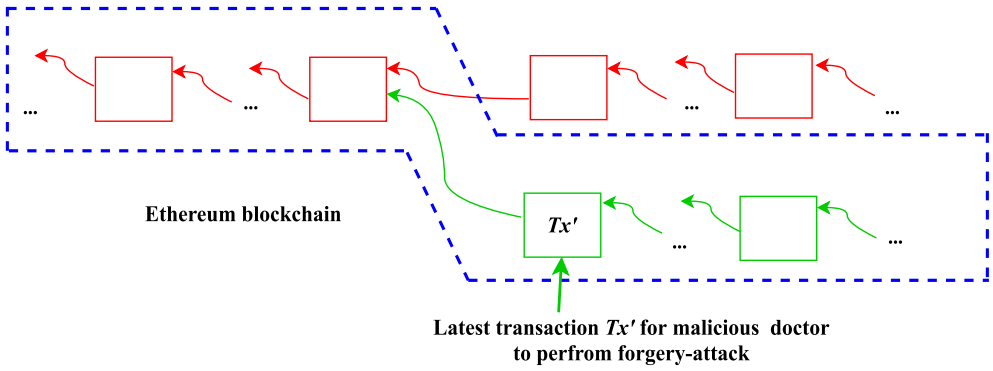


**Fig. 5** Forgery-attack by the malicious CM.

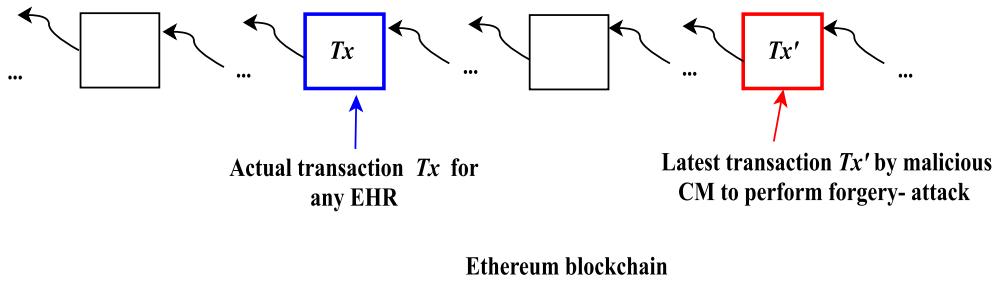


**Fig. 6** Forgery-attack by the semi-trusted doctors.

The defined game begins at the round where a predetermined initial block identified by 0 is uploaded and asserts whenever the chain length extends at $(n+\Delta)$, where $(n,\Delta) \in Z$. The $n^{th}$-block is the successive block and its extract decides whether the adversary wins or not. The given block can be modified only by forking. So, the consensus algorithm of the blockchain does not go to the real blockchain but to the latest longest chain of the adversary. The malicious doctor or adversary may try to exploit this by attempting to generate a fork to modify an unfavorable successive block. So, we consider $\wp$ be the probability for uniformly distributed extract value $(m \in \imath)$ to satisfy $\chi(m) = 1$ and simply defined $\wp = \frac{|\chi(m)=1|}{|\imath|}$. However, all the possible outputs of the game can be divided into an infinite set of disjoint events $(E_f, f)$ which defines the valid blocks. Further, we also assume that $\omega$ - represent the adversary or malicious doctor wins the game. So, the events $E_f \cap \omega$requires the final block $B$ in the longest chain, which satisfies $\chi(m) = 1$. So, the winning probability $P_{A_I}$ can be defined as

$$P_{A_I} = \sum_{f \geqslant 0} Pr(E_f \cap \omega) = \sum_{f \geqslant 0} (\lambda(1-\wp))^f \wp$$

where $\lambda$ = represents the probability that the adversary or malicious doctor will complete the next oracle with a valid block. As for the winning condition $\lambda \geqslant 0, \wp \leqslant 1$, this is significantly better than $\wp$. For instance, on the winning condition, $\lambda = 1/4, \wp = 1/2, P_{A_I}$ is increased from 0.50 to 0.572. Therefore, by performing a hard-fork with limited power and a hash rate is less than 51% of the overall network's hash rate, none of the adversary or malicious doctors can break the validity of the Ethereum blockchain [47].

• **Case 2.** For the next case, any malicious doctor generates an EHR and outsources it to CSP. However, later, he/she tries to convince that the EHRs are generated by any other doctor.

**Proof:** Note that in the proposed model, before outsourcing the EHRs, the patients don't need to authenticate their generated EHRs. They only need to authenticate their assigned doctors before the scheduled diagnosis. Thus, the patient $(Pt)$ creates a warrant $(WA_{Pt}, Wt_{Pt})$ to meet the doctor $(Dc)$. This warrant consists of the details of the assigned doctors, i.e., doctor's ID $(ID_{Dc})$ with their diagnosing-time and other related information. However, this warrant is also computationally secure until CDH-problem is hard to solve in probabilistic polynomial time with non-negligible probability [44]. Thus, it is computationally infeasible to perform such an attack for any malicious doctor.

**Theorem 4.** The proposed model also ensures the timeliness of the outsourced EHRs.

**Proof.** The timeliness of EHRs is represented in the associated timestamp-value, i.e., $Tm$ of transaction in the block of the blockchain. Notably, in the proposed model, each of the outsourced EHRs always relate to one transaction in the blockchain. Thus, after the transaction is chained into the blockchain, anyone can easily extract the generation time of EHR from the timestamp-value, i.e., $Tm$ of transaction.

## 7. Performance analysis

In this section, the performance of the proposed model is analyzed. Also, the experimental analysis of the proposed model is evaluated by comparing it with other existing methodologies [21,22,31,49]. The whole set-up is designed on systems with Ubuntu 16.04 LTS, Intel(R) Xeon(R) E-2124G CPU@ 3.40 GHz ×64 - based 64 bit processor, 32 GB DDR3 RAM, 2 TB SATA memory with 64 MB, Amazon cloud infrastructure [45] set-up for cloud environment, and PBC-(0.5.14) library used to implement all cryptographic operations and OpenSSL crypto-library to complete 80-bit security token $(l)$. Initially, the fundamental conditions are based on an MNT 159 curve, which has a 160-bit supersingular secure elliptic curve $E_n$ over 512-bit finite field $F_q$. As described earlier, Ethereum blockchain used as the platform for the public blockchain. We deployed the local version of Ethereum blockchain, i.e., Ganache client Geth 1.9.0 to design the proposed model. Moreover, the proposed model is implemented on Amazon cloud infrastructure, where various virtual machines Ubuntu 16.04 LTS with other required specifications are configured as the Patients, Hospital with doctors, Cloud service provider (CSP), and CM, respectively. Further, each of the Amazon cloud infrastructure's function communicate with Ganache via **web3js** API [46]. The **web3js** library is lightweight java library that establishes a IPC or HTTP connection to interact with Ganache.

Moreover, we also set-up IPFS on the Amazon cloud infrastructure to establish a decentralized storage platform.

Presently, IPFS infrastructure can be built on the top of the Amazon cloud infrastructure. Along with these, a java library is also implement to upload the outsourced EHRs to IPFS storage on cloud environment.

### 7.1. Communication overhead

The communication overhead between the doctors and CSP depends on the size of the generated EHRs. Thus, in the communication analysis of the proposed model, we would exclude the communication overhead of outsourcing the EHRs. The communication cost for the hospital $Ht$ consists two sub-parts; the first one is related to communication between patient $Pt$ to make an appointment, and next one is for assigning the doctor $Dc$ for corresponding patient $Pt$. Like as the communication cost for the patient $Pt$ consists of two sub-parts; the first one is related to make an appointment to the hospital $Ht$, and the next one is to meet with doctor $Dc$. So, the communication overhead to make appointment and meet the doctor $Dc$ is only $2(|p| + |k|)$, where $|p|$ is the size of the identity of the hospital and doctor, and $k|$ is the size of human-memorisable-password.

Moreover, at the CSP side, the communication cost includes three sub-parts; the first one is for extracting the hash-value of $\varphi$ - successive blocks from Ethereum blockchain to check the correctness of $\varphi$ - successive blocks from the doctor's side. The next one is related to upload the generated transaction $(Tx)$ to Ethereum blockchain. Finally, the last one is related to store the outsourced EHRs in distributed IPFS-nodes. So, the communication overhead on the CSP side includes only $|n|$, where $|n|$ is the total number of transactions uploaded to blockchain. The other models [21,31,49] bear high communication overhead due to generations of challenges and proofs to complete auditing procedures from TPA to CSP and CSP and TPA, respectively. However, the proposed model only extracts the latest block from the blockchain and checks the warrant's validity only to verify the integrity of the outsourced EHRs. So, the proposed model only requires $(|n| + 2)G_1$. Further, Fig. 7 illustrates the communication overhead on all the related entities. So, Fig. 7 describes that all the entities in the proposed model would bear significant communication overhead. Fig. 8 shows that the comparison with the single outsourcing of EHR by a single doctor for a single patient and the batch EHRs outsourcing of EHR by multiple doctors for multiple patients simultaneously is more advantageous for the CM on communication overhead. Further, Fig. 8 describes that the comparison with single EHR outsourcing, the batch outsourcing of EHRs reduces the communication cost of CM. This analysis also shows that the communication costs are adequate in practice.

### 7.2. Computational overhead

We evaluate the computational overhead of the proposed model in terms of basic cryptographic operations during the experimental analysis. Initially, the hospital $Ht$ and patient $Pt$ entities interact to generate the treatment_key (TK). With the treatment_key (TK), the patient $(Pt)$ gets the appointment. Then, the patient $Pt$ can generate a warrant $(WA_{Pt})$ and meet the doctor $Dc$. For this instance, the corresponding computational cost is calculated as; $(n+4)E_p + 6 \cdot Mult_{G_1} + 2 \cdot Hash_{Z_p} + (n+2)Hash_{G_1}$ , where, $n$ - represents the number of doctors, $E_p$ - exponent operation in $G_1$, $Mult_{G_1}$ - point multiplication operation in $G_1$, $Hash_{Z_p}$ - Hash function in $Z_p$, $Hash_{G_1}$ - Hash function in $G_1$. Afterwards, the doctor $Dc$ generates the corresponding EHR and outsource it to the CM at CSP, which includes the encrypted EHR with corresponding signature_value $(\sigma)$. For this instance, the corresponding computational cost is calculated as follows; $n \cdot Hash_{G_1} + (2n+1)Mult_{G_1} + n \cdot E_{nc} + n \cdot Pt_A + (n+1)E_p$, where, $E_{nc}$- encryption operation to encrypt the EHRs, $Pt_A$-point addition operation in $G_1$. Finally, the CM stores it and uploads the transactions $(Tx)$ to Ethereum blockchain. So, on the CSP side, the computational costs are; $2n \cdot (Po)_{G_2} + n \cdot Hash_{G_1} + n \cdot Hash_{Z_p}$ -, where $Po_{G_2}$ pairing operation in $G_2$. In Fig. 9, we describe the computational cost on the hospital $Ht$ and the CSP. Further, the experimental results illustrate that the proposed model takes only 1 s to outsource the generated EHRs

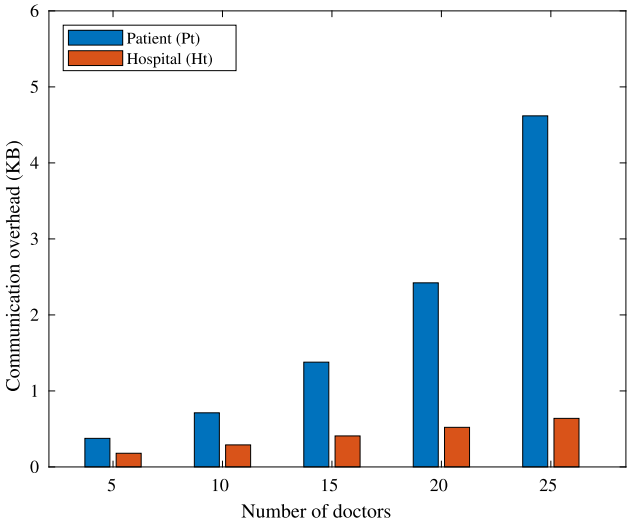


**Fig. 7** Communication overhead on the hospital ($Ht$ and patient $Pt$).

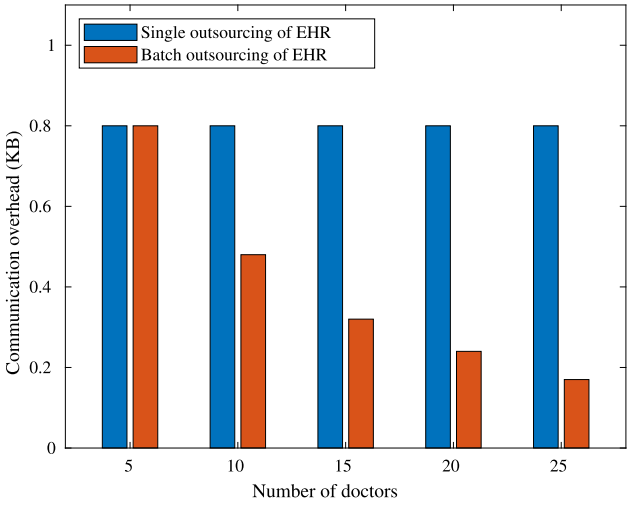


**Fig. 8** Communication overhead of single outsourcing of EHR and batch outsourcing of EHR.

to CSP for the doctor *Dc*, who is properly managing with a large number of computers. The computational delay on the CM for single outsourcing of EHR of a single patient by a single doctor is within 16 ms. However, in real scenario, the CM would support multiple doctors simultaneously. In this scenario, the batch outsourcing of EHRs would dramatically reduce the computational overhead. Fig. 10 depicts the experimental evaluation of the comparison between single outsourcing of EHR and batch outsourcing of EHRs in terms of computational delay.

Specifically, to illustrate the experimental analysis, we analyze the generation of signature_value on $G_1$ for all generated EHRs and EHR's outsourcing from doctor $(ID_{DC_i})$ to IPFS nodes on CSP by following tests; firstly, the doctor $(ID_{DC_i})$ generate the EHR with a fixed size, which is limited by $|G_1|$. Without the loss of generality, suppose the doctor $(ID_{DC_i})$ generates EHRs ranging in size from 100 MB to 1000 MB, increasing incrementally from 100 MB in every test case; also creates the associated signature_value for all generated EHRs. Fig. 11 illustrates that the computational overhead for generation signature_value grows linearly with the size of EHRs. Moreover, we also perform some required tests to analyze the computational overhead for the auditing procedure in the proposed model and other existing models [21,22,31,49]. In [21,31,49], TPA randomly selects a number and subset of challenged EHRs ($l$), then CSP generates proofs as a response. So, TPA challenges different subsets of EHRs with a continuous increment of 100 MB sets of outsourced EHRs in each test. However, in [22] and the proposed model, TPA only extracts the block at the latest height of blockchain and verifies the total number of generated transactions to the number of EHRs generated by a doctor $(ID_{DC_i})$. Finally, TPA verifies the validity of the warrant $(Wt_{P_t})$ and signature_value $(\sigma)$ to check the integrity of outsourced EHRs. So, the proposed model and [22] only requires constant computational overhead for auditing procedures to verify the integrity of outsourced EHRs. Fig. 12 and Fig. 13 shows the overall computational overhead of the existing models - [21,22,31,49] and proposed model to perform the auditing procedure at TPA and CSP, respectively. Moreover, the complete analysis of the computational overhead of the EHR outsourcing and storage procedure is described in Table 3. On the other hand, Table 3. also shows

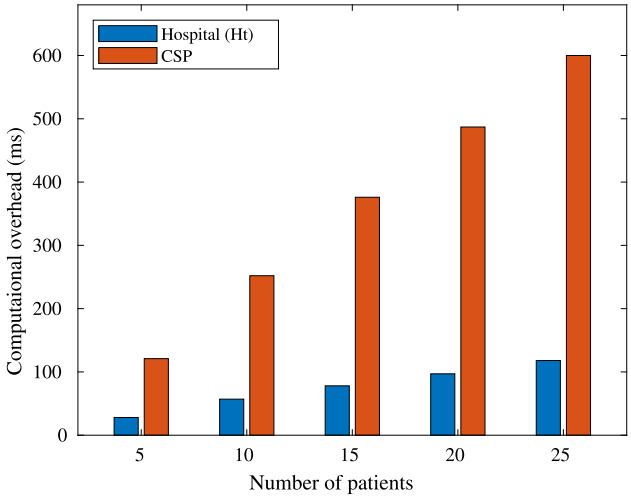


**Fig. 9** Computational overhead on (*Ht* and CSP).

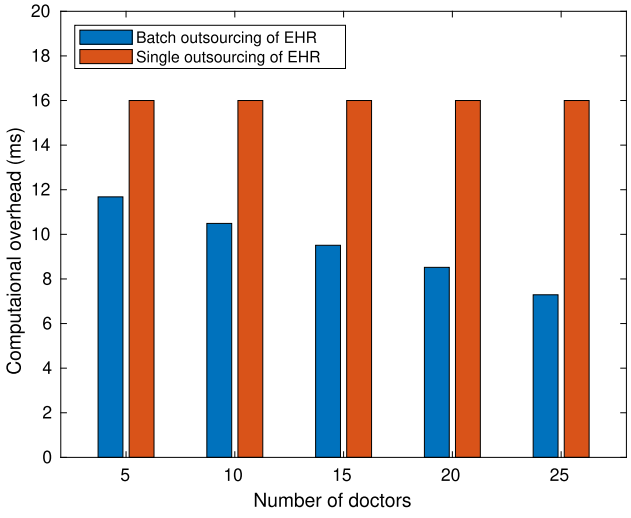


**Fig. 10** Computational overhead on the CM w.r.t. the number of doctors.

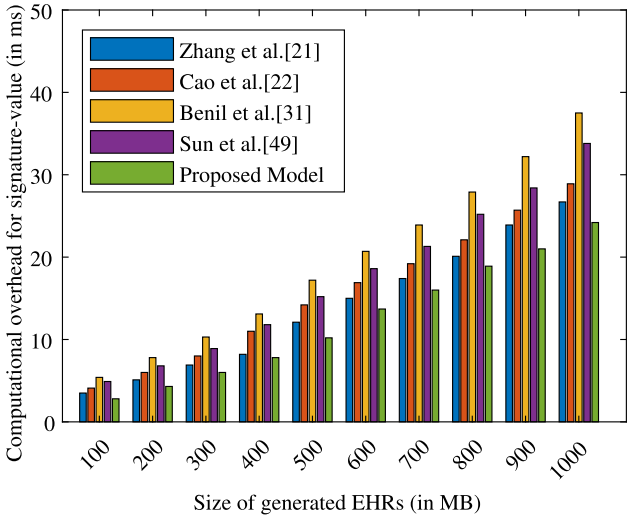


**Fig. 11** Computational overhead of signature-value generation.

that the computational overhead of the proposed model is lower than the models described by [21,22,31,49], especially in terms of pairing operation $(Po)_{G_2}$. This lower computational cost occurs due to reducing the number of cryptographic operations.

For concrete experimental analysis, we demonstrate the system performance in terms of latency and throughput of transactions; also capacity and size of blocks over the blockchain as follows;

### *7.2.1. Transaction latency*

Transaction latency refers to the overall execution time throughout the blockchain, including the transaction broadcasting time and execution time of the consensus algorithm.

$$T_{LT} = (T_{CT} \times T_{NT}) - T_{ST}$$

where $T_{LT}, T_{CT}, T_{NT}, T_{ST}$ are transaction-latency, transaction-conformation-time, network-threshold, and transaction-submission-time, respectively.

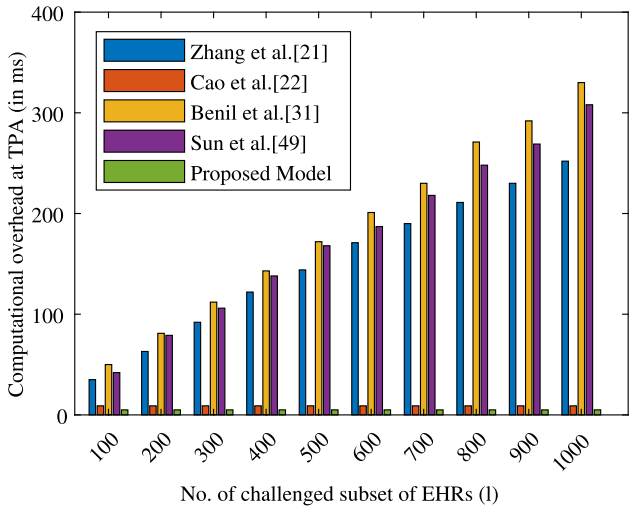


**Fig. 12** Computational overhead of TPA for auditing procedure.

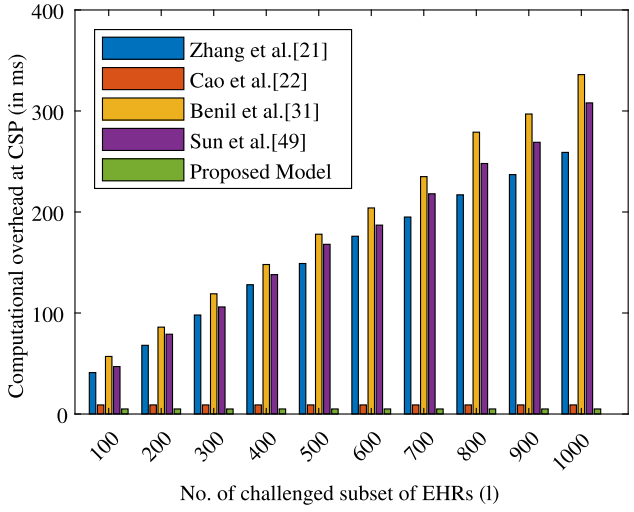


**Fig. 13** Computational overhead of CSP for auditing procedure.

The average transaction latency is obtained by continuously changing the transaction send rate from 100 to 1000 transactions-per second (tsp) to outsourced the EHRs. Fig. 14 illustrates that the proposed EHR storage blockchain framework's average transaction latency is considerably increased until the transaction-send-rate (tps) reaches up to 650 tsp. The transaction latency of the proposed EHR storage blockchain framework grows linearly as, after the transaction send-rate (tps) of 650 tsp, the number of transaction generation requests continuously increases. So, the transaction send rate of 650 tsp is considered the average transaction send rate of the proposed model. Also, the size of generated EHRs varies and can be enormous. Further, the encryption operation increases EHR's size, making storage of these EHRs over blockchain a costlier operation. The proposed model employs only IPFS hash-value of encrypted EHRs stored in the blockchain to address these issues. Fig. 15, illustrates the latency result of uploading the EHRs on IPFS nodes at CSP. This figure shows that the latency of uploading the EHRs generally increased with the EHR size. The maximum time related to uploading any generated EHR on IPFS nodes is 27 s with an EHR of size 1000 MB.

### 7.2.2. Transaction throughput

Transaction throughput refers to the number of successful transactions completed on the blockchain during a specified time slot.

$$Transaction - throughput = \frac{Successful - transactions}{Time(sec)}$$

Note that the invalid transactions should be excluded from the set of total transactions to get the successful transactions. Fig. 16 shows that the average transaction throughput continuously rises as the transaction send-rate rises until it achieves the threshold value of transaction send-rate 700 tps. Further, in the optimal case, the average transaction throughput is considered to be 700 tps; after that, the throughput decreases by increasing the transaction send rate.

### 7.2.3. Block capacity and block size

The proposed model's performance analysis also considers the capacity of blocks and the number of transactions over the blockchain. The baseline for analysis has been established at 100 blocks having EHR transactions. The number of blocks has continuously increased by 100 to 1000 blocks. Let us suppose that the average EHR size is 25 KB, and the block header's size can be as standard 80 bytes. Further, the standard size of a block is 1 MB, so 100 blocks with blockchain storage can contain 4096 transactions. Moreover, we consider that a block contains only 1000 transactions, which is continuously increasing by 1000 to illustrate the analysis of the blockchain height. This analysis shows that approximately 78 blocks can be present over the blockchain.

Thus, the experimental analysis shows that the proposed model is efficient and can be easily implemented in practice.

## 7.3. Security comparison

Table 4 compares the proposed model to with other existing schemes [18,21,22,24,25,27,29,31,32] earlier reviewed in the

**Table 3** Comparison of computational overheads.

| Schemes | Overall computation overhead |
|---|---|
| Cao et al. [22] | $2n \cdot (PO)_{G_2} + (2n+3) \cdot Pt_A + (3n+4)Mult_{G_1} + (2n+1)Hash_{G_1} + (n+1)Hash_{Z_p} + (2n+3)E_p$ |
| Zhang et al. [21] | $3 \cdot (PO)_{G_2} + (n-1)Pt_A + (n+4)Mult_{G_1} + n \cdot Hash_{G_1} + n \cdot Hash_{Z_p} + (n+1)E_p$ |
| Sun et al. [49] | $(2n+1)(PO)_{G_2} + (n+1)Pt_A + (n+2)E_p + (n+2)Hash_{G_1} + (n+1)Hash_{Z_p} + (4n+2)Mult_{G_1}$ |
| Benil et al. [31] | $(4n)(PO)_{G_2} + (3n+1)Pt_A + (2n+1)E_p + (6n)Hash_{G_1} + (n)Hash_{Z_p} + (4n+3)Mult_{G_1}$ |
| Proposed | $2 \cdot (PO)_{G_2} + n \cdot Pt_A + (2n+6)Mult_{G_1} + (3n+2)Hash_{G_1} + (n+2)Hash_{Z_p} + (2n+4)E_p$ |

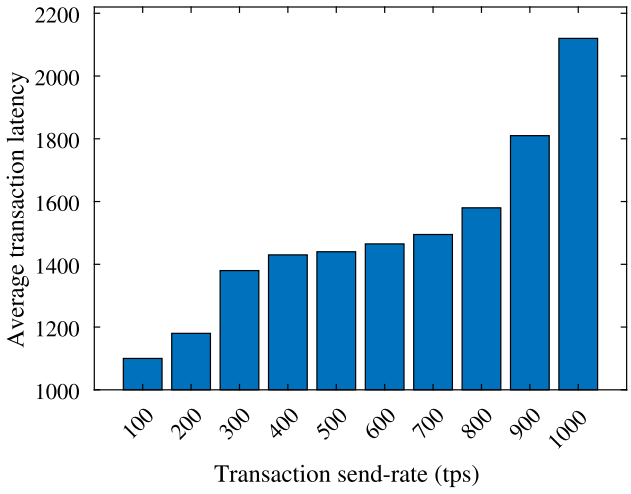


**Fig. 14** Average transaction latency of the proposed model.

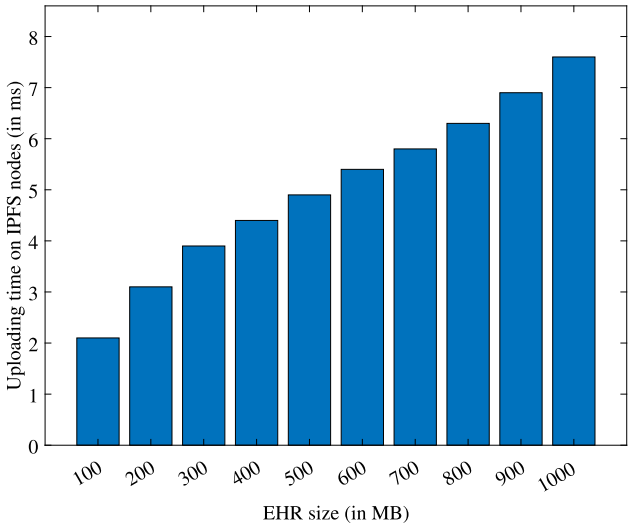


**Fig. 15** Uploading time of outsourced EHRs in IPFS nodes.

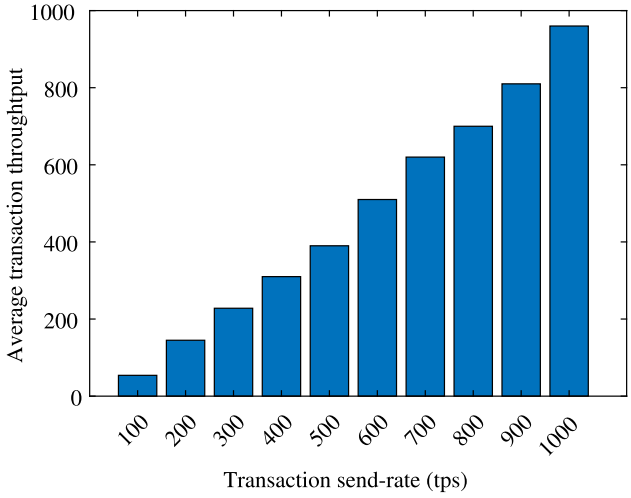


**Fig. 16** Average transaction throughput of the proposed model.

related work section in terms of features and functionality. The main comparison focuses on the standard security (Universal Composability (UC)), whether the proposed model follows it. The standard security (Universal Composability (UC)) model is a significant component of computationally intractability, associated with data integrity and confidentiality. So, Table 4 illustrates that the proposed model only follows it to achieve complete intractability against forgery and modification attack to ensure data integrity with proper probabilistic concrete proof. In addition, Considering the impersonation attack, the work of [22], and the proposed model employs the secure password-based authenticated key-exchange protocol to ensure security against password guessing attack by an external adversary. Meanwhile, the infeasibility against collusion attacks of the proposed model also guarantees none of the malicious entities can perform collusion attacks to break the security of the model. Moreover, only the work [21], and the proposed model can simultaneously handle multiple outsourcing of generated EHRs and multiple auditing of outsourced EHRs to support efficient batch outsourcing and auditing, respectively. Table 4 shows the comparison between the proposed model and other existing models in terms of storage feature of the proposed model. The storage of EHRs directly into blockchain database, causing a significant scalability issues. It could also be implemented in a centralized or local database, which raises issues of privacy and confidentiality. So, to address the scalability and storage completeness issues, the proposed model stores the outsourced EHRs in decentralized storage (i.e., IPFS) and store only their hash-values in blockchain. Therefore, as observed from Table 4, the proposed model can significantly support the standard security features.

## 8. Discussion

The proposed model illustrates how to integrate blockchain technology to design a tamper-proof storage model for EHR. As the EHR of the patients is confidential and highly sensitive, it is vital to construct a tamper-proof EHR storage model via immutable blockchain technology to secure sensitive healthcare records. Blockchain technology has been considered a powerful technique to address the integrity and confidentiality-related security issues with cloud-based EHR storage models. The proposed model differs from existing EHR storage models in that we employ a peer-to-peer decentralized storage system, i.e., IPFS, to avoid a single point of failure for EHR storage. It makes the proposed model an ideal EHR storage model in a cloud environment where proper availability is highly demanded. It does not need to be dependent on any centralized storage server. Moreover, the other key feature is its ability always to establish a secure communication channel to take infeasible online appointments for patients. In the implementation, we are also conscious of the public blockchain's limitations towards hard-fork to reverse any transactions after the latest DAO hacks.

As described in Section 6 (Security analysis), the proposed model outperforms the existing models significantly in terms of proper security against - collusion, modification, and forgery attacks with the standard security model (UC model). The security analysis follows the computational intractability of hard CDH and DDH assumptions [41]. Furthermore, Section 7

**Table 4** Comparative analysis in the security of different schemes and proposed model.

| Schemes | Public auditable | Standard Security model (UC model) | Storage Completeness | Resistance against Impersonation attack | Batch outsourcing and auditing | Resistance against Collusion attack | Database storage | Scalability |
|---|---|---|---|---|---|---|---|---|
| Zhang et al. [18] | ✓ | X | ✓ | X | X | X | Cloud storage | X |
| Zhang et al. [21] | ✓ | X | ✓ | X | ✓ | X | Cloud database | X |
| Cao et al. [22] | ✓ | X | ✓ | ✓ | X | X | Blockchain database | X |
| Zaabar et al. [24] | ✓ | X | ✓ | X | X | X | Orbit DB with IPFS | X |
| Yu et al. [25] | ✓ | X | ✓ | X | X | X | Blockchain database | X |
| Li et al. [27] | ✓ | X | ✓ | X | X | X | Blockchain database | X |
| Hussien et al. [29] | ✓ | X | ✓ | X | X | X | Blockchain with IPFS | ✓ |
| Benil et al. [31] | ✓ | X | ✓ | X | X | X | Blockchain database | X |
| Ghayvat et al. [32] | ✓ | X | ✓ | ✓ | X | X | Blockchain database | ✓ |
| Proposed | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | Blockchain with IPFS | ✓ |

(Performance analysis) shows that the proposed model incurs low computational overhead with various standard security features is another major goal. The merits of the proposed model come at the cost of extra communication overhead for appointment procedures in Section 5.1.2. The reason is that the proposed model employs an intractable password-based authenticated key-exchange protocol to establish a secure channel to take appointments for patients. This password-based authenticated key exchange protocol incurs significant extra communication overhead to the proposed model. Besides, based on the listed features of the proposed model, we can discern that a suitable blockchain-based tamper-proof EHR storage model will be implemented in the near future.

## 9. Conclusion and future work

In this paper, a blockchain-based tamper-proof EHR storage in the cloud environment has been proposed. This strategy ensures the proper infeasibility of the outsourced EHRs in distributed IPFS nodes. On the other hand, the proposed model also assures the proper computationally unforgeability of outsourced EHRs. Any malicious doctor who generates and outsources the EHRs may collude with the CSP to forge the outsourced EHRs. The proposed model is implemented on the Ethereum blockchain, where the generated EHRs are integrated into a transaction of the Ethereum blockchain. Thus, the integrity of the outsourced EHRs follows the computational intractability of Ethereum, which ensures the timeliness of outsourced EHRs. So, anyone can efficiently extract the generation time of EHRs. The security analysis illustrates that the proposed model is computationally unforgeable to numerous attacks in distributed IPFS nodes at CSP for EHR storage. The comprehensive numerical evaluation and comparison of experimental results may be utilized to validate the proposed model's overall performance, demonstrating that the proposed model is much practical and effective in terms of computation and communication overhead.

In a real-world scenario, it is not reasonable to store the outsourced EHRs on cloud storage and respective transactions on a blockchain network for a long-time. So, we will further explore the scalable consortium blockchain architecture to support the deletion of blocks (i.e., deletion of outsourced EHRs after some time). This scalable consortium blockchain architecture may support effective storage management on the blockchain with proper immutability and high scalability. In addition, efficient decentralized auditing may also be needed to handle a single point of failure for auditing procedures at a single TPA. Therefore, a potential direction of future work is to design a deletable blockchain-based immutable EHR storage in the cloud environment with efficient decentralized auditing.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

This research work is supported by Indian Institute of Technology (ISM), Dhanbad, Govt. of India. The authors wish to express their gratitude and heartiest thanks to the Depart-

ment of Computer Science & Engineering, Indian Institute of Technology (ISM), Dhanbad, India for providing their research support.

## References


[1] Blesson Varghese, Rajkumar Buyya, Next generation cloud computing: New trends and research directions, Future Gener. Comput. Syst. 79 (2018) 849–861.

[2] Tatiana Ermakova et al, Security and privacy requirements for cloud computing in healthcare: elicitation and prioritization from a patient perspective, ACM Trans. Manage. Inform. Syst. (TMIS) 11 (2) (2020) 1–29.

[3] Fangjian Gao, Ali Sunyaev, Context matters: A review of the determinant factors in the decision to adopt cloud computing in healthcare, Int. J. Inf. Manage. 48 (2019) 120–138.

[4] Chuan Zhang et al, LPTD: Achieving lightweight and privacy-preserving truth discovery in CIoT, Future Gener. Comput. Syst. 90 (2019) 175–184.

[5] Chuan Zhang et al, PPDP: An efficient and privacy-preserving disease prediction scheme in cloud-based e-Healthcare system, Future Gener. Comput. Syst. 79 (2018) 16–25.

[6] Xuguang Wu et al, Secure Personal Health Records Sharing Based on Blockchain and IPFS, in: Chinese Conference on Trusted Computing and Information Security, Springer, Singapore, 2019.

[7] Arshdeep Bahga, Vijay K. Madisetti, A cloud-based approach for interoperable electronic health records (EHRs), IEEE J. Biomed. Health Inform. 17 (5) (2013) 894–906.

[8] Wenting Shen et al, Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage, IEEE Trans. Inf. Forensics Secur. 14 (2) (2018) 331–346.

[9] Benet, Juan. Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561 (2014).

[10] Nishara Nizamuddin, Haya R. Hasan, Khaled Salah, IPFS-blockchain-based authenticity of online publications, in: International Conference on Blockchain, Springer, Cham, 2018.

[11] Sheng Cao, Xiaosong Zhang, Xu. Rixin, Toward secure storage in cloud-based eHealth systems: a blockchain-assisted approach, IEEE Network 34 (2) (2020) 64–70.

[12] Maithilee Joshi, Karuna Pande Joshi, Tim Finin, Delegated authorization framework for EHR services using attribute based encryption, IEEE Trans. Serv. Comput. (2019).

[13] Ismail Keshta, Ammar Odeh, Security and privacy of electronic health records: Concerns and challenges, Egypt. Inform. J. 22 (2) (2021) 177–183.

[14] Tehsin Kanwal, Adeel Anjum, Saif UR Malik, Abid Khan, Muazzam A. Khan, Privacy preservation of electronic health records with adversarial attacks identification in hybrid cloud, Comput. Stand. Interfaces 78 (2021) 103522.

[15] Rumenigue Hohemberger, Cinara Ewerling, Felipe da Rosa, Rubin Pfeifer, Rodrigo Mello, Paulo da Rosa, Silas Severo, Arthur de Souza, Francisco Lorenzon, Marcelo Caggiani Luizelli, Fábio Diniz Rossi, An approach to mitigate challenges to the electronic health records storage, Measurement 154 (2020) 107424.

[16] Wei-Bin Lee, Chien-Ding Lee, A cryptographic key management solution for HIPAA privacy/security regulations, IEEE Trans. Inf Technol. Biomed. 12 (1) (2008) 34–41.

[17] Sun, Jinyuan, et al. HCPP: Cryptography based secure EHR system for patient privacy and emergency healthcare. 2011 31st International Conference on Distributed Computing Systems. IEEE, 2011.

[18] Yinghui Zhang, Dong Zheng, Robert H. Deng, Security and privacy in smart health: Efficient policy-hiding attribute-based access control, IEEE Internet Things J. 5 (3) (2018) 2130–2145.

[19] Jianghong Wei, Xiaofeng Chen, Xinyi Huang, Hu. Xuexian, Willy Susilo, RS-HABE: Revocable-storage and hierarchical attribute-based access scheme for secure sharing of e-health records in public cloud, IEEE Trans. Dependable Secure Comput. 18 (5) (2019) 2301–2315.

[20] Rupa Ch et al, Robust Cyber-Physical System Enabled Smart Healthcare Unit Using Blockchain Technology, Electronics 11 (19) (2022) 3070.

[21] Xiaojun Zhang et al, CIPPPA: Conditional Identity Privacy-Preserving Public Auditing for Cloud-Based WBANs against Malicious Auditors, IEEE Trans. Cloud Comput. (2019).

[22] Sheng Cao et al, Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain, Inf. Sci. 485 (2019) 427–440.

[23] Rahul Mishra, Dharavath Ramesh, Damodar Reddy Edla, Lianyong Qi, DS-Chain: A secure and auditable multi-cloud assisted EHR storage model on efficient deletable blockchain, J. Industr. Inform. Integr. (2022) 100315.

[24] Hongjiao Wu, Ashutosh Dhar Dwivedi, Gautam Srivastava, Security and privacy of patient information in medical systems based on blockchain technology, ACM Trans. Multimedia Comput., Commun., Appl. (TOMM) 17 (2s) (2021) 1–17.

[25] Keping Yu et al, Efficient and privacy-preserving medical research support platform against COVID-19: a blockchain-based approach, IEEE Consum. Electron. Magaz. 10 (2) (2020) 111–120.

[26] Haiping Huang, Fu. Xiang Sun, Peng Zhu Xiao, Wenming Wang, Blockchain-based eHealth system for auditable EHRs manipulation in cloud environments, J. Parallel Distrib. Comput. 148 (2021) 46–57.

[27] Fengqi Li, Kemeng Liu, Lupeng Zhang, Sikai Huang, Wu. Qiufan, EHRChain: A Blockchain-based EHR System Using Attribute-Based and Homomorphic Cryptosystem, IEEE Trans. Serv. Comput. (2021).

[28] Shiyu Li, Yuan Zhang, Xu. Chunxiang, Nan Cheng, Zhi Liu, Xuemin Sherman Shen, BESURE: Blockchain-Based Cloud-Assisted eHealth System with Secure Data Provenance, in: In 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS), IEEE, 2021, pp. 1–6.

[29] Hassan Mansur Hussien, Sharifah Md Yasin, Nur Izura Udzir, Mohd Izuan Hafez Ninggal, Blockchain-based access control scheme for secure shared personal health records over decentralised storage, Sensors 21 (7) (2021) 2462.

[30] Gaby G. Dagher et al, Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology, Sustain. cities Soc. 39 (2018) 283–297.

[31] T. Benil, J.J.C.N. Jasper, Cloud based security on outsourcing using blockchain in E-health systems, Comput. Netw. 178 (2020) 107344.

[32] Muhammad Junaid Iqbal et al, RThreatDroid: A Ransomware Detection Approach to Secure IoT Based Healthcare Systems, IEEE Trans. Network Sci. Eng. (2022).

[33] Armknecht, Frederik, et al. Outsourced proofs of retrievability. Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. 2014.

[34] Hovav Shacham, Brent Waters, Compact proofs of retrievability, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, Berlin, Heidelberg, 2008.

[35] Yuan, Jiawei, and Shucheng Yu. Proofs of retrievability with public verifiability and constant communication cost in cloud. Proceedings of the 2013 international workshop on Security in cloud computing. 2013.

[36] Petar Maymounkov, David Mazieres, Kademlia: A peer-to-peer information system based on the xor metric, International Workshop on Peer-to-Peer Systems, Springer, Berlin, Heidelberg, 2002.

[37] Douglas R. Stinson, Some observations on the theory of cryptographic hash functions, Des. Codes Crypt. 38 (2) (2006) 259–277.
[38] Gavin Wood, Ethereum: A secure decentralised generalised transaction ledger, Ethereum project yellow paper 151 (2014) (2014) 1–32.
[39] Yong Ho Hwang, Dae Hyun Yum, Pil Joong Lee, EPA: An efficient password-based protocol for authenticated key exchange, in: Australasian Conference on Information Security and Privacy, Springer, Berlin, Heidelberg, 2003.
[40] Jonathan Katz, Rafail Ostrovsky, Moti Yung, Efficient password-authenticated key exchange using human-memorable passwords, in: International Conference on the Theory and Applications of Cryptographic Techniques, Springer, Berlin, Heidelberg, 2001.
[41] Dan Boneh, Ben Lynn, Hovav Shacham, Short signatures from the Weil pairing, J. Cryptol. 17 (4) (2004) 297–319.
[42] Adem Efe Gencer et al, Decentralization in bitcoin and ethereum networks, in: International Conference on Financial Cryptography and Data Security, Springer, Berlin, Heidelberg, 2018.
[43] Limin Shen et al, A secure and efficient id-based aggregate signature scheme for wireless sensor networks, IEEE Internet of Things Journal 4 (2) (2016) 546–554.
[44] Dan Boneh, Emily Shen, Brent Waters, Strongly unforgeable signatures based on computational Diffie-Hellman, International Workshop on Public Key Cryptography, Springer, Berlin, Heidelberg, 2006.
[45] Amazon Web Services (AWS)–Cloud Computing Services. [Online]. Available: https://aws.amazon.com/.
[46] Lightweight Java and Android Library for Integration With Ethereum Clients. [Online]. Available: https://github.com/web3j/web3j.
[47] Cécile Pierrot, Benjamin Wesolowski, Malleability of the blockchain's entropy, Cryptogr. Commun. 10 (1) (2018) 211–233.
[48] Xiaodong Yang et al, Medical data sharing scheme based on attribute cryptosystem and blockchain technology, IEEE Access 8 (2020) 45468–45476.
[49] Jin Sun et al, Blockchain-based secure storage and access scheme for electronic medical records in IPFS, IEEE Access 8 (2020) 59389–59401.
[50] Usharani Chelladurai, Seethalakshmi Pandian, A novel blockchain based electronic health record automation system for healthcare, J. Ambient Intell. Human. Comput. (2021) 1–11.
[51] Tiantian Li et al, FAPS: A fair, autonomous and privacy-preserving scheme for big data exchange based on oblivious transfer, Ether cheque and smart contracts, Inf. Sci. 544 (2021) 469–484.