ASTES

*Advances in Science, Technology and Engineering Systems Journal Vol. 7, No. 2, 179-190 (2022)*

*www.astesj.com*

*Special Issue on Multidisciplinary Sciences and Engineering*

**ASTESJ**

**ISSN: 2415-6698**

# Cloud-Based Hierarchical Consortium Blockchain Networks for Timely Publication and Efficient Retrieval of Electronic Health Records

Alvin Thamrin, Haiping Xu*, Rui Ming

*Computer and Information Science Department, University of Massachusetts Dartmouth, Dartmouth, MA 02747, USA*

ARTICLE INFO

*Article history:*
*Received: 30 February, 2022*
*Accepted: 17 April, 2022*
*Online: 22 April, 2022*

*Keywords:*
*Hierarchical blockchain*
*Cloud Computing*
*Timely publication*
*Electronic health records*
*Consensus mechanism*

ABSTRACT

*Blockchain technology is seeing a trend of popularity and adoption in many different application areas. One such area is healthcare, as there is a need to develop a system that can reliably store and share electronic health records (EHRs) among hospital-based health facilities. In this paper, we present a cloud-based hierarchical consortium blockchain framework for storing and sharing EHRs in a scalable, secure, and reliable manner. The framework enables data sharing between local hospital blockchain networks (HBNs) through high-level blockchain networks, namely, city blockchain networks (CBNs) and a state blockchain network (SBN). To support the timely publication of EHRs in HBNs, we adopt a temporary and permanent block scheme in hospital blockchains. In addition, we develop role-based access control (RBAC) policies for data authorization and procedures for concurrent search and retrieval of EHRs across cities and states. The experimental results show that our proposed approach is feasible and supports timely publication and efficient retrieval of EHRs in cloud-based hierarchical blockchain networks.*

## 1. Introduction

Blockchain technology was originally proposed in 2008 as a decentralized and distributed digital ledger mechanism for the peer-to-peer electronic cash system called Bitcoin [1]. A blockchain stores data in blocks that are cryptographically chained together in the form of a linked list. Thus, blocks can be used to store and record transactions in a tamper-proof and immutable manner. Unlike public blockchains, a consortium blockchain is defined as a permissioned blockchain, and access to it is usually restricted to a specific number of "permissioned" nodes [2]. In recent years, the popularity of consortium blockchain has increased due to its potential use in many different application areas, including healthcare [3], [4]. A consortium blockchain-based system can be implemented within the healthcare domain to enable and support the storage and sharing of healthcare data among health institutions or hospitals. In our earlier work, we introduced a cloud-based blockchain solution for storing and sharing electronic health records (EHRs) while enabling data accessibility, redundancy, and security on a local scale [5]. This solution allows storing big data, such as EHRs with multimedia files, in a cloud-based blockchain, while storing their metadata in a lite blockchain for efficient information retrieval. However, due to the big data involved, the solution can only be effective when implemented on a small/local scope. This is because the growth potential of the blockchain increases dramatically with the large number of hospitals participating in the network. This can lead to a very unsustainable expansion of the blockchain in terms of size, which constitutes a major scalability issue.

In this paper, we present a cloud-based hierarchical consortium blockchain framework to address the above scalability issue. The framework consists of three layers of blockchain networks, namely hospital blockchain networks (HBNs), city blockchain networks (CBNs), and a state blockchain network (SBN). An HBN is designated as a blockchain network at the first layer and is shared by hospitals that are geographically close to each other in a local area or a city. To simplify matters, in this paper we use the term *city* to refer to a city, a local area or any form of governmental jurisdiction below the state level. A CBN is designated as a blockchain network at the second layer. Unlike an HBN, a CBN is shared by all cities located within a state as participants. Each city in a CBN is also connected to an HBN as the network regulator, which allows agents from different HBNs within the same state to communicate with each other for data sharing purposes. Finally, An SBN is designated as a blockchain network at the third layer. The SBN is shared by all states located within a country. Each state in the SBN is also connected to a CBN and acts as the network regulator of the CBN. The SBN is designed to allow agents from

*Corresponding Author: Haiping Xu, University of Massachusetts Dartmouth, Dartmouth, MA 02747, Email: hxu@umassd.edu

different CBNs across the country to communicate with each other for data sharing purposes, similar to the sharing relationship between a CBN and the HBNs connected under it. The implementation of these network layers enables all hospital peers across the country to communicate and share data with each other in a scalable, secure, and reliable manner.

Another challenging issue we face concerns the publishing of EHRs to the blockchain in a timely and space-efficient manner. Whenever data are made available, they can either be published to the blockchain immediately in the form of block records stored in a block, or they can be accumulated until the block contains a sufficient number of block records to be published. The first method excels in terms of timeliness but is inefficient in terms of space/memory usage because this method generates many blocks containing a single or very few records. On the other hand, the second method is more spatially efficient compared to the first one because fewer and denser blocks are generated; however, it has a significant drawback in terms of timeliness that may affect the effectiveness of a blockchain-based system for storing and sharing EHRs. In this paper, we present an approach that facilitates the timely and space-efficient publication of new block records using a temporary and permanent block scheme. As demonstrated in previous work [6], a new block record can be published immediately in a temporary block after being approved using a temporary block consensus mechanism. Once a predefined number of temporary blocks have been published, they can be merged into a permanent block and published to the blockchain.

This work significantly extends our previous proposed framework for healthcare data storage using hierarchical cloud-based blockchains. In our previous work [7], we focused on the structural design of the storage system and did not fully consider the scalability of HBNs with many peers and the timely publication of EHRs in HBNs. To address these issues, we now limit the number of hospital super-peer agents in an HBN to reduce the redundancy of big data storage. Furthermore, by using a temporary and permanent block scheme, EHRs can be efficiently published in HBNs. Finally, in previous work [7], new access control policies need to be established and approved after the search results are returned. In this work, we require that access control policies be established prior to the doctor's visits. Thus, the search and retrieval steps of EHRs can be combined to achieve an efficient information search and retrieval process.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 presents a cloud-based hierarchical consortium blockchain framework. Section 4 introduces the block structures and the processes of generating and publishing new blocks in different blockchains. Section 5 describes the search and retrieval process of EHRs in details. Section 6 presents the case studies and their analysis results. Section 7 concludes the paper and mentions future work.

## 2. Related Work

There are various studies and explorations on blockchain technology to develop a decentralized storage and sharing system for the healthcare sector [3]-[5]. Blockchain technology has been shown to be a viable technology as it allows sharing of medical data among approved healthcare providers while maintaining patient privacy [8]. Further research has also addressed the challenge of storing big data such as images and videos in the blockchain; however, these studies have typically utilized off-chain approaches to store big data, rather than on-chain solutions. In [9], the authors proposed a storage model based on blockchain and InterPlanetary File System (IPFS) to store transactions efficiently in blockchain. In their design, the actual patient reports are stored in distributed off-chain storage using IPFS, while the blockchain stores only hash values of the reports, thereby reducing the overall block size in the blockchain. In [10], the authors developed a decentralized and permissible blockchain-based application for storing and accessing satellite task-scheduling schemes using the Hyperledger Fabric framework. They used IPFS for off-chain storage to reduce the asset size of the transaction and increase the transaction throughput of the network. In [11], the authors introduced a video surveillance storage and sharing system using blockchain technology. In their approach, videos received from the camera are encrypted and stored off-chain through distributed IPFS system with their metadata stored in the blockchain. Some researchers also proposed a secure data sharing solution for sensitive financial data using blockchain and proxy re-encryption technology [12]. Access control rules, hash values, and storage addresses of financial data are stored in the blockchain, while the actual financial data are stored off-chain in distributed databases. In [13], the authors proposed a blockchain framework using an attribute-based cryptosystem for the development of a secure EHR storage and sharing system. In their approach, large-scale medical data are stored in the cloud and the blockchain stores only the metadata of EHRs. Unlike these approaches, our cloud-based blockchain solution enables all healthcare data, including multimedia files, to be stored in the blockchains. Thus, our on-chain data storage approach provides the benefits of a complete blockchain storage solution in terms of data immutability, integrity and availability.

There are other studies focusing on the design of new blockchain architecture, which are summarized below. In [14], the authors proposed Fortified-Chain, a decentralized EHR and blockchain-based distributed data storage system (DDSS). They designed a global DDSS network that facilitates communications between local DDSS networks consisting of hospitals and third-party health services that store patient medical data. In [15], the authors developed a simplified version of a scalable blockchain architecture for sharing EHRs among patients, healthcare professionals and health institutions. In their approach, each health facility implements a local blockchain network connected to a global blockchain system to allow interaction between different health institutions. Some researchers also studied and introduced a blockchainless approach based on directed acyclic graphs (DAGs) for trusted public construction bidding to ensure fairness in the bidding process [16]. The DAG-based approach differs from the traditional blockchain because in the chainless approach, the DAG links the transaction containing its parents, documents, and a list of transaction signatures to other transactions through a less complex verification process. In a more recent effort, researchers designed a Compacted DAG-based blockchain protocol (CoDAG), used in the field of Industrial Internet of Things (IIoT) [17]. They developed protocols and algorithms to secure the network and confirm transactions within a specified time. The aforementioned blockchain-based approaches either use off-chain storage, e.g.,

[14] and [15], or do not address scalability issues, e.g., [16] and [17]. Unlike the above approaches, our novel cloud-based hierarchical blockchain architecture not only supports on-chain storage of big data, but also allows interaction and data sharing between peers located in different cities and states. Since EHRs from different cities are stored in different HBNs, our cloud-based hierarchical blockchain approach provides a scalable solution for storing sensitive information and big data in a nationwide network of connected consortium blockchains.

Previous research efforts on implementing access control mechanisms in blockchain networks have focused on preventing unauthorized access to confidential data stored in the blockchains. In [4], the authors proposed MeDShare, a blockchain-based system that enables peer-to-peer medical data sharing in a trustless environment. They used smart contracts and access control mechanisms to monitor and track the behavior of storing data in the blockchain. If any form of data permission violation is detected, the system revokes the access rights of the offending user. In [18], the authors proposed a Blockchain-as-a-Service based solution for Health Information Exchange (BaaS-HIE) activities to deal with security issues including patient privacy, integrity of medical records, and fine-grained access control. Their approach involves the use of a private blockchain based on the Ethereum protocol and smart contracts as access control management for medical records. In a similar way, other researchers designed an access control mechanism on managing user access to ensure efficient and secure sharing of EHRs on mobile devices by leveraging smart contracts on the Ethereum blockchain [19]. In [20], the authors proposed the use of blockchain and edge nodes to facilitate attribute-based access control and storage of EHR data. They used smart contracts to enforce access control of EHR data stored in off-chain edge nodes. In their subsequent work, encryption for data stored at the edge nodes was further developed [21]. The multi-authority attribute-based encryption (ABE) scheme and attribute-based multi-signature (ABMS) scheme were used to encrypt the EHR data stored at the edge nodes and verify users' signatures, respectively. In contrast to the above work, our approach involves the implementation of different scopes of role-based access control (RBAC) policies that restrict user access to various healthcare facilities in different cities and states. We define three layers of the networks, each implementing its own RBAC policies – local hospital-wide policies, city-wide policies, and statewide policies, respectively. As a result, our approach provides a more comprehensive and reliable mechanism than other methods because it is designed to work in a much larger environment.

## 3. A Framework for Hierarchical Blockchains

The framework for cloud-based hierarchical consortium blockchain networks consists of three layers. As shown in Figure 1, these layers are the *Hospital Layer*, the *City Layer*, and the *State Layer*, which contain multiple HBNs, multiple CBNs, and an SBN, respectively. An HBN in the hospital layer covers multiple hospitals from the same city or local area, represented by hospital super-peer agents $\beta_{HOSs}$ or hospital regular-peer agents $\beta_{HREPs}$. A CBN in the city layer covers multiple cities from the same state. A city super-peer agent $\beta_{CIT}$ acts as a representative of a city and a network regulator for the HBN belonging to the city. Finally, an SBN in the state layer covers all states of a country. A state super-peer agent $\beta_{STA}$ acts as a representative of a state and a network regulator for the CBN belonging to the state.

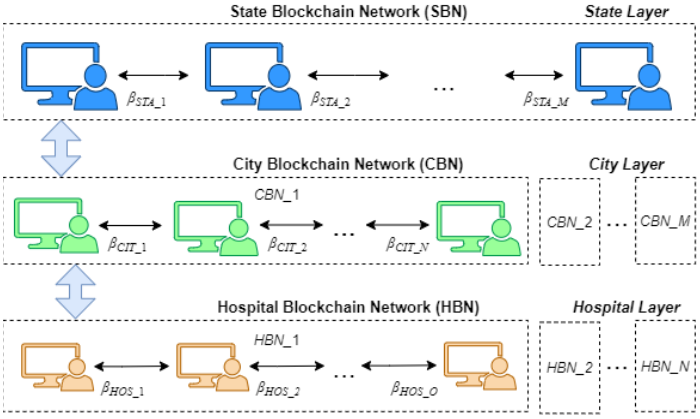


Figure 1: The Architecture of Cloud-Based Hierarchical Consortium Blockchains

To enable big data storage in blockchain networks, a cloud-based blockchain scheme is implemented in the hospital layer, i.e., HBNs. Unlike previous work [5], which requires all participating hospitals in an HBN to implement cloud-based blockchains, in this study, the HBN consists of three types of agents, namely the hospital super-peer agents $\beta_{HOSs}$, representing designated hospitals, hospital regular-peer agents $\beta_{HREPs}$, representing general hospitals, and regular-peer agents $\beta_{REPs}$, representing end users including doctors, nurses, and patients. We define general hospitals as those that do not have the required infrastructure to implement cloud-based blockchain storage or choose not to do so. Figure 2 shows an example of an HBN where hospital *A* and *B* are designated hospitals that offer private cloud services, while hospital *C* is a general hospital that do not provide such services.

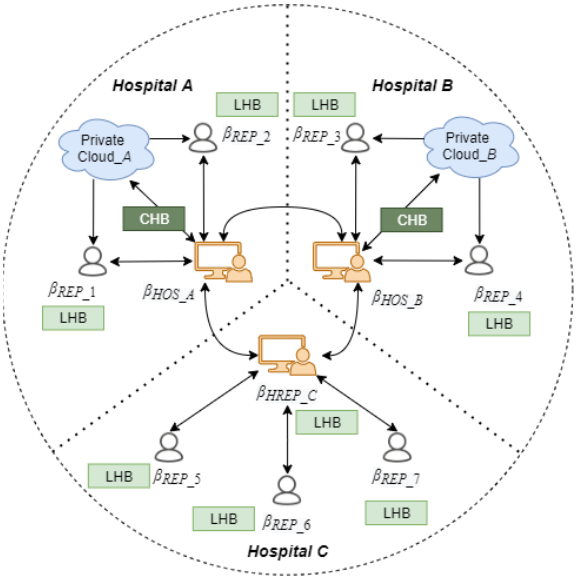


Figure 2: The Relationships Between Participants in an HBN

As shown in Figure 2, a cloud-based hospital blockchain (CHB) is implemented in the designated hospitals with their private clouds. A CHB is managed by a hospital super-peer agent $\beta_{HOS}$ and stores all data, including EHRs in multimedia file format. To avoid excessive redundancy of big data in an HBN, we limit the number of hospital super peer agents in an HBN to no more

than 6-10. A lite hospital blockchain (LHB) is implemented on the server of a general hospital, managed by a hospital regular-peer agent $\beta_{HREP}$, or on the local machine of an end user, managed by a regular-peer agent $\beta_{REP}$. An LHB stores all data in its corresponding CHB, except for big data such as multimedia files, for which only their metadata are stored in the LHB. Access to confidential data, i.e., a patient's EHRs, stored in a CHB is managed by a hospital super-peer agent $\beta_{HOS}$, while access to confidential data stored in an LHB is managed by either a hospital super-peer agent $\beta_{HOS}$ or a hospital regular-peer agent $\beta_{HREP}$.

Figure 3 shows the general blockchain structure and the similarity between CHB and LHB. Let the length of a LHB and its corresponding LHB be $h$. A cloud-based block $CB_i$ and a lite block $LB_i$, where $1 \leq i \leq h$, contain the same information except for the multimedia files. This scheme allows an end user or a general hospital to use the metadata stored in its LHB to submit a request to a relevant hospital super-peer agent $\beta_{HOS}$ through its regular-peer agent $\beta_{REP}$ or hospital regular-peer agent $\beta_{HREP}$ and retrieve the corresponding multimedia files stored in the CHB.

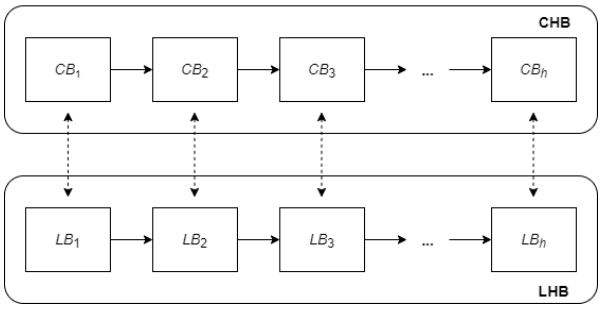


Figure 3: The General Blockchain Structure of a CHB and an LHB

To support the timely publication of EHRs in an HBN, we introduce a temporary and permanent block scheme based on earlier work [6]. Due to the need to publish EHRs, including their associated multimedia files in a timely manner, temporary blocks are only included in the hospital layer of our cloud-based hierarchical blockchain networks; however, they are not required in the city and state layers, as access control policies and access records do not need to be published immediately. Figure 4 shows an example of an CHB with temporary and permanent blocks.

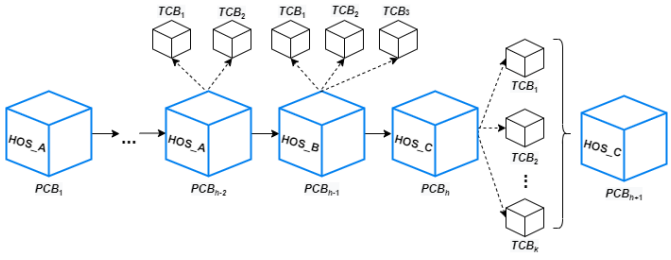


Figure 4: A CHB with Temporary and Permanent Blocks

As shown in Figure 4, a block $PCB_i$, where $i$ is the height of the block in the blockchain, is a cloud-based permanent block. To support efficient information retrieval, a *PCB* contains only EHRs and other related records from the same hospital. On the other hand, a block $TCB_j$, where $j$ denotes the order in which the temporary blocks are attached to a *PCB* according to their publishing time, is a cloud-based temporary block. To support the timely publication of block records, a *TCB* stores only one record from a hospital and must be attached to the latest *PCB* published for the same hospital. As shown in the figure, the latest *PCB* for a hospital can be attached by multiple *TCB*s that are numbered in the order in which they appear. Note that the LHB corresponding to a CHB shares the same structure but have different notations, i.e., permanent lite block *PLB* and temporary lite block *TLB*.

Since a *TCB* contains only one block record, whenever a block record is generated and submitted to a super-peer agent, the agent can immediately publish the block record to the blockchain as a temporary block through the temporary block generation process. Meanwhile, a permanent block stores multiple block records as in a typical blockchain. Once enough *TCB*s are generated and published to the blockchain by a hospital super-peer agent, the agent can consolidate them into a new *PCB* through the permanent block generation process. For example, as in Figure 4, when the number or total size of *TCB*s attached to $PCB_h$ reaches a threshold, the agent $\beta_{HOS\_C}$ merges the list of *TCB*s and forms a new cloud-based permanent block $PCB_{h+1}$ for publishing. When $PCB_{h+1}$ is published, all *TCB*s attached to $PCB_h$ are removed from the blockchain. Note that other *TCB*s that are attached to *PCB*s other than $PCB_h$ will remain in the blockchain until they are merged into new *PCB*s.

## 4. Publication of New Blocks in the Blockchain Networks

An HBN, a CBN or the SBN maintains its own blockchain, namely hospital blockchain, city blockchain or state blockchain, respectively. Blockchain networks of the same type, such as two HBNs, are independent, but they can communicate through a higher-level blockchain network, e.g., a CBN if the two HBNs belong to the same state, or the SBN if the two HBNs are in different states. Hospital, city, and state blockchains can store different types of block records for different purposes. In this section, we describe the types of block records used in different blockchains and the procedures for generating and publishing new blocks in different types of blockchains.

### *4.1. Hospital Block and its Block Record Types*

There are four different types of block records that can be used in a CHB or an LHB, namely $HR_{UPR}$, $HR_{ACP}$, $HR_{MER}$, and $HR_{AR}$. To simplify matters, we define a hospital blockchain as a general term that can refer to a CHB or an LHB. We now describe the four types of block records as follows.

- $HR_{UPR}$ is a record that stores the account information and user profile of an end user, represented by regular-peer agent $\beta_{REP}$. An $HR_{UPR}$ is defined as a 6-tuple ($I$, $N$, $R$, $U$, $S$, $T$), where $I$ is the identification of $\beta_{REP}$ in the HBN; $N$ is the full name of $\beta_{REP}$; $R$, $U$ and $S$ are $\beta_{REP}$'s private key, public key and secret symmetric key, respectively; and $T$ is the timestamp when the $HR_{UPR}$ is created. Whenever a new user joins the HBN or an existing user's profile is updated, a new $HR_{UPR}$ is created.
- $HR_{ACP}$ is a record that stores access control policies and is used to conduct permission checks on requests to access EHRs stored at hospitals within the same city. An $HR_{ACP}$ is defined as a triple ($P$, $H$, $T$), where $P$ is a set of policies; $H$ is a set of hospital where the policies are executed; and $T$ is the timestamp when the policies are created.
- $HR_{AR}$ is a record that stores information on access requests to a patient's EHRs stored at hospitals within the same city where

the patient resides. $HR_{AR}$ is created as a log of access requests for accountability purposes. An $HR_{AR}$ is defined as 4-tuple ($N$, $D$, $O$, $T$), where $N$ is the request number; $D$ is the detail of the request; $O$ is the outcome of the request; and $T$ is the timestamp when the request is created.

- $HR_{MER}$ is a record that stores medical information, including patient reports and metadata for any related multimedia files generated after a doctor's visit. An $HR_{MER}$ is defined as 5-tuple ($I$, $H$, $X$, $M$, $T$), where $I$ are the identifications of all peers involved in the doctor's visit, including the patient, the nurse and the doctor; $H$ is the name of the hospital where the patient visited; $X$ includes a summary of the visit and any text-based medical data; $M$ is the metadata of any multimedia files generated after the doctor's visit; and $T$ is the timestamp when the $HR_{MER}$ record is created.

Since both permanent and temporary blocks in a CHB may contain multimedia files, the blocks *PCB* and *TCB* consist of two major components: the block component and the multimedia file component. Figure 5 shows the block structure of a new temporary cloud-based block $TCB_j$ with three sections in the block component and one section in the multimedia file component. These are header, hospital block records, verification information, and multimedia files in an EHR.

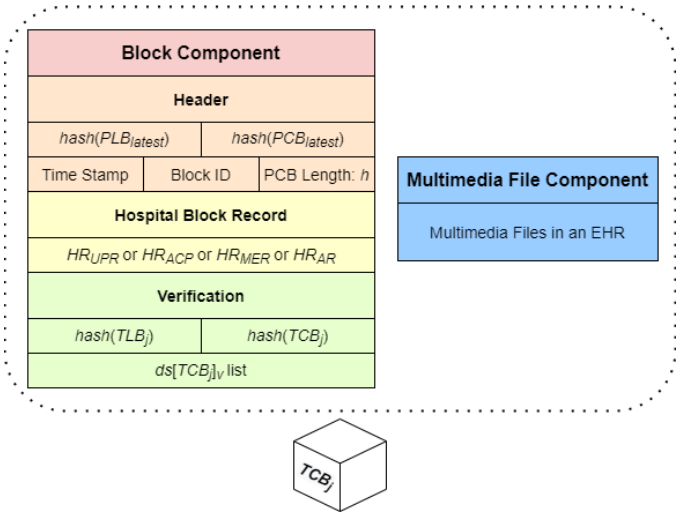


Figure 5: The Structure of a New Temporary Cloud-Based Block $TCB_j$

As shown in Figure 5, the header section contains the hash values of the latest *PLB* and *PCB*, published by the same hospital super-peer agent who generates $TCB_j$, the timestamp when $TCB_j$ was created, the block ID, and the length $h$ of the current blockchain. The hospital block records section contains only a single block record of $HR_{UPR}$, $HR_{ACP}$, $HR_{MER}$, or $HR_{AR}$ as $TCB_j$ is a temporary block. Consequently, the multimedia file section can only store multimedia files from one doctor's visit, if any, while their metadata is recorded and stored in the relevant $HR_{MER}$ in the hospital block record section. Lastly, the verification section contains the hash values of the current block, including the hash value of the header and hospital block records, denoted as $hash(TLB_j)$, and the hash value of the header, hospital block records and the multimedia files, denoted as $hash(TCB_j)$. The verification section also contains a list of digital signatures $ds[TCB_j]_v$, where each peer $v$ is an agent $\beta_{HOS}$ who approves $TCB_j$ during the temporary block consensus process. Note that the structure of the temporary lite block $TLB_j$ is similar to that of $TCB_j$ but does not include the multimedia file component.

Figure 6 shows the block structure of a new permanent cloud-based block $PCB_{h+1}$. The block structure of *PCB* is similar to that of *TCB*, but a *PCB* can accommodate multiple block records and EHRs in its hospital block records section and multimedia file component, respectively.

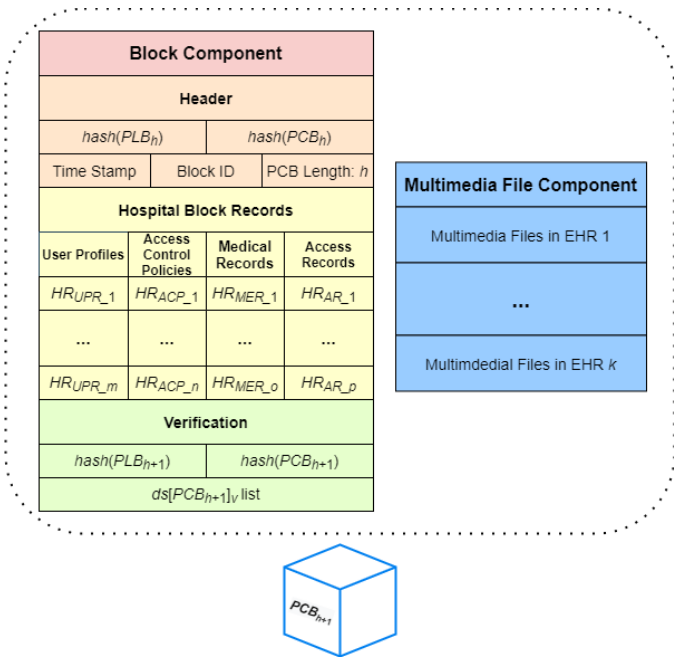


Figure 6: The Structure of a New Permanent Cloud-Based Block $PCB_{h+1}$

In a *PCB*, the verification section contains the hash values of the permanent lite block $PLB_{h+1}$ and the permanent cloud-based block $PCB_{h+1}$. It also contains a list of digital signatures $ds[PCB_{h+1}]_v$, where each peer $v$ is an agent $\beta_{HOS}$, who approves $PCB_{h+1}$ during the permanent block consensus process. Note that while a new *TCB* is attached to the latest *PCB*, published by the hospital super-peer agent who generates the *TCB*, a new *PCB* must be attached to the last *PCB* of the cloud-based blockchain, i.e., $PCB_h$, where $h$ is the height of the current blockchain. As with temporary blocks, the structure of a permanent lite block $PLB_{h+1}$ is similar to that of $PCB_{h+1}$ except for the inclusion of the multimedia file component in $PCB_{h+1}$.

### *4.2. Hospital Temporary and Permanent Block Generation*

Let hospital super-peer agent $\beta_{HOS\text{-}\Psi}$ be the one who creates a new temporary cloud-based block $TCB_j$. Algorithm 1 shows how the new block $TCB_j$ is generated by agent $\beta_{HOS\text{-}\Psi}$. According to the algorithm, agent $\beta_{HOS\text{-}\Psi}$ first creates an empty temporary cloud-based block $TCB_j$. All attributes in $TCB_j$'s header section are then created and added. These include the hash values of the latest permanent blocks, previously published by $\beta_{HOS\text{-}\Psi}$, i.e., $hash(PCB_{latest})$ and $hash(PLB_{latest})$, the timestamp when $TCB_j$ is created, $TCB_j$'s block ID, and the blockchain length $h$. After that, $\beta_{HOS\text{-}\Psi}$ processes the hospital block record $\varphi$ given in the input list accordingly. If $\varphi$ is an $HR_{UPR}$ and $\varphi.S$ is *null*, it indicates that $\varphi$ records the account information and user profile of a new end user. In this case, a secret symmetric key is automatically generated and added it to $\varphi.S$. Then $\varphi$ is encrypted using the public key of $\beta_{HOS\text{-}}$

$_\psi$ and added to the hospital block record section of $TCB_j$. If $\varphi$ is a medical block record $HR_{MER}$ and a list $\rho$ of multimedia files is included, $\beta_{HOS\text{-}\psi}$ encrypts the files in $\rho$ using the associated patient's secret symmetric key retrieved from the patient's latest $HR_{UPR}$. The encrypted files are then added to the multimedia file component of $TCB_j$. The metadata of the encrypted files are also recorded and added to $\varphi.M$ of the medical block record. Finally, $\varphi$ itself is encrypted, except for $\varphi.M$, before it is added to the hospital block record section of $TCB_j$. Note that if $\varphi$ is an $HR_{ACP}$ or $HR_{AR}$, it is added directly to the hospital block record section of $TCB_j$ in plaintext. Once the header section and hospital block record section are established, $\beta_{HOS\text{-}\psi}$ calculates the hash values $hash(TLB_j)$ and $hash(TCB_j)$, as well as the digital signature $ds[TCB_j]_\psi$ using $hash(TCB_j)$. All these elements are then added to the verification section of $TCB_j$. Note that while not shown in Algorithm 1, a new temporary lite block $TLB_j$ can be created by simply removing the multimedia file component from $TCB_j$.

**Algorithm 1: Generating a New Temporary Block $TCB_j$**

**Input:** A hospital block record $\varphi$ containing record $HR_{UPR}$, $HR_{ACP}$, $HR_{AR}$, or $HR_{MER}$, and an optional list $\rho$ of multimedia files.
**Output:** A new temporary cloud-based block $TCB_j$.

1. Create an empty temporary cloud-based block $TCB_j$
2. Verify and add $hash(PCB_{latest})$, $hash(PLB_{latest})$, time stamp, block ID and current blockchain length $h$ to the header section of $TCB_j$
3. **if** $\varphi$ is an $HR_{UPR}$ and $\varphi.S$ is *null* // indicates a new end user
4. Generate a secret symmetric key, add it to $\varphi.S$, and encrypt $\varphi$
5. **else if** $\varphi$ is an $HR_{MER}$ and $\rho$ is not empty
6. Encrypt the multimedia files in $\rho$
7. Add the encrypted files to the multimedia file section of $TCB_j$
8. Add the metadata of $\rho$ to $\varphi.M$ and encrypt $\varphi$, except for $\varphi.M$
9. Add $\varphi$ to the hospital block record section of $TCB_j$
10. Calculate the hash values $hash(TCB_j)$ and $hash(TLB_j)$
11. Add the hash values to the verification section of $TCB_j$
12. Create digital signature $ds[TCB_j]_\psi$ using $hash(TCB_j)$
13. Add $ds[TCB_j]_\psi$ to the $ds[TCB_j]_v$ list in the verification section
14. **return** $TCB_j$

Once enough *TCB*s are generated and published to the blockchain by $\beta_{HOS\text{-}\psi}$, the *TCB*s can be consolidated into a new permanent block *PCB* through a permanent block generation process. Algorithm 2 shows how a new permanent cloud-based block $PCB_{h+1}$ is generated by agent $\beta_{HOS\text{-}\psi}$. According to the algorithm, agent $\beta_{HOS\text{-}\psi}$ first creates an empty permanent cloud-based block $PCB_{h+1}$. All attributes in $PCB_{h+1}$'s header section, including $hash(PCB_h)$ and $hash(PLB_h)$, the timestamp, the block ID, and the blockchain length $h$, are then created and added. For each temporary block $\tau$ in the temporary block list $\Xi$, $\beta_{HOS\text{-}\psi}$ verifies it using information stored in $\tau$'s header and the verification section and transfers all relevant information from the hospital block record section of $\tau$ to the hospital block records section of $PCB_{h+1}$ as a new block record. If $\tau$ contains a block record $HR_{MER}$ and a list of encrypted multimedia files $\rho$, $\beta_{HOS\text{-}\psi}$ moves files in $\rho$ to the multimedia file component of $PCB_{h+1}$ and adds the metadata of $\rho$ to the relevant block record $HR_{MER}.M$. This ensures that all previously stored information in the temporary blocks from the list $\Xi$ is transferred to $PCB_{h+1}$. Finally, $\beta_{HOS\text{-}\psi}$ calculates the hash values $hash(PCB_{h+1})$ and $hash(PLB_{h+1})$, as well as the digital signature $ds[PCB_{h+1}]_\psi$ using $hash(PCB_{h+1})$. All these elements are then added to the verification section of $PCB_{h+1}$. Similar to the generation of $TLB_j$, a new permanent lite block $PLB_{h+1}$ can be created by simply removing the multimedia file component from $PCB_{h+1}$.

**Algorithm 2: Generating a New Permanent Block $PCB_{h+1}$**

**Input:** A list of blocks $\Xi$ containing $k$ temporary cloud-based blocks.
**Output:** A new permanent cloud-based block $PCB_{h+1}$.

1. Create an empty permanent cloud-based block $PCB_{h+1}$
2. Verify and add $hash(PCB_h)$, $hash(PLB_h)$, time stamp, block ID, and current blockchain length $h$ to the header section of $PCB_{h+1}$
3. **for** each temporary cloud-based block $\tau$ in $\Xi$
4. Verify $\tau$ and add all relevant parts from the hospital block record section in $\tau$ to the hospital block records section in $PCB_{h+1}$
5. **if** $\tau$ contains $HR_{MER}$ and a list of encrypted multimedia files $\rho$
6. Add files in $\rho$ to $PCB_{h+1}$'s multimedia file component
7. Add the metadata of $\rho$ to the corresponding $HR_{MER}.M$
8. Calculate hash values $hash(PCB_{h+1})$ and $hash(PLB_{h+1})$
9. Add the hash values to the verification section of $PCB_{h+1}$
10. Create digital signature $ds[PCB_{h+1}]_\psi$ using $hash(PCB_{h+1})$
11. Add $ds[PCB_{h+1}]_\psi$ to the $ds[PCB_{h+1}]_v$ list in the verification section
12. **return** $PCB_{h+1}$

### *4.3. City and State Block and their Block Record Types*

Unlike the hospital blockchain, there is only one variant of the city and state blockchains due to the absence of regular peers and big data. Thus, implementing cloud-based versions of city and state blockchains is not necessary. For city blockchain, there are two types of block records that can be stored in the blockchain. These are city-wide record for access control policies $CR_{ACP}$ and city-wide access record $CR_{AR}$. $CR_{ACP}$ is a record that stores the access control policies implemented in a CBN and enforced by the relevant city super-peer agent $\beta_{CIT}$. $CR_{ACP}$ has the same structure as $HR_{ACP}$, except that the $CR_{ACP}.H$ contains additional information such as the names of cities and hospitals where the policies are enforced. $CR_{ACP}$ is created to check any requests regarding access to patient EHRs stored in HBNs across cities within the same state. On the other hand, $CR_{AR}$ is a record that stores information on access requests to patient EHRs in hospitals across cities within the same state. The structure of a $CR_{AR}$ is also similar to that of an $HR_{AR}$.

For state blockchain, a state block shares the same structure as that of a city block and stores statewide records for access control policies $SR_{ACP}$ and statewide access record $SR_{AR}$. $SR_{ACP}$ is a record that stores the access control policies implemented in the SBN and enforced by the relevant state super-peer agent $\beta_{STA}$. $SR_{ACP}$ is established to check for any access requests to patient EHRs stored in HBNs across states; while $SR_{AR}$ is a record that stores information on access requests to patient EHRs in hospitals across states. Figure 7 shows the structure of a new city or state block $B_{h+1}$ in a city or state blockchain. From the figure, we can see that block $B_{h+1}$ consists of only one component as city and state blockchains do not store EHRs. There are three sections present in block $B_{h+1}$, namely header, state or city block records, and the verification section. The header section contains the previous city or state block's hash value $hash(B_h)$, the timestamp when $B_{h+1}$ is created, the block ID of $B_{h+1}$, and the current blockchain length $h$. The city or state records section contains a list of block record $CR_{ACP}$ and/or $CR_{AR}$, or $SR_{ACP}$ and/or $SR_{AR}$, respectively. The verification section contains the hash values of $B_{h+1}$ and a list of

digital signatures $ds[B_{h+1}]_v$, where each peer $v$ is a city or state super-peer agent who approves $B_{h+1}$ during the consensus process.

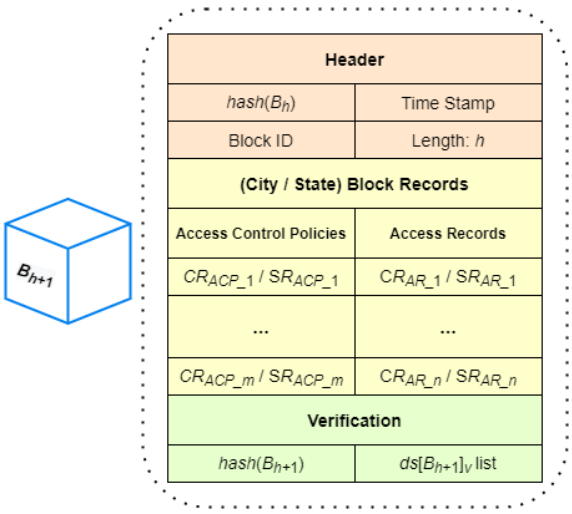


Figure 7: The Structure of a New City or State Block $B_{h+1}$

### 4.4. City and State Block Generation

The process of generating a new city or state block is similar to that of generating a lite hospital block, although it is simpler due to the absence of big data and end users. Let the city super-peer agent $\beta_{CIT\text{-}\Psi}$ be the one who creates the new city block $B_{h+1}$. Algorithm 3 shows the procedure for generating $B_{h+1}$, which is then approved and added to the city blockchain through a city block consensus process.

**Algorithm 3: Generating a New City Block $B_{h+1}$**

**Input:** A list of city block records $\Phi$ containing $CR_{ACP}$ and/or $CR_{AR}$
**Output:** A new city block $B_{h+1}$

1. Create an empty city block $B_{h+1}$
2. Verify and add $hash(B_h)$, time stamp, block ID, and current blockchain length $h$ to the header section of $B_{h+1}$
3. **for** each record $\varphi$ in the list $\Phi$ of city block records
4. **if** $\varphi$ is an $CR_{ACP}$
5. Add $\varphi$ to the city block records section of $B_{h+1}$
6. **else**
7. Encrypt $\varphi$ and add it to the city block records section
8. Calculate $hash(B_{h+1})$ and add it to the verification section
9. Create digital signature $ds[B_h]_\Psi$ using $hash(B_{h+1})$
10. Add $ds[B_h]_\Psi$ to the $ds[B_h]_v$ list in the verification section
11. **return** $B_{h+1}$

According to the algorithm, agent $\beta_{CIT\text{-}\Psi}$ first creates an empty city block $B_{h+1}$. All attributes in $B_{h+1}$'s header section are then created and added. These include the previous block's hash value $hash(B_h)$, the timestamp when $B_{h+1}$ is created, the block ID, and the blockchain length $h$. After that, $\beta_{CIT\text{-}\Psi}$ processes all records in the city block record list $\Phi$ accordingly before they are added to the city block records section of $B_{h+1}$. If a city block record $\varphi$ is a $CR_{ACP}$, it is simply added to $B_{h+1}$'s city block records section without being encrypted. Afterwards, $\beta_{CIT\text{-}\Psi}$ calculates $hash(B_{h+1})$ and $ds[B_{h+1}]_\Psi$ before adding them to the verification section of $B_{h+1}$. Note that the algorithm for generating a new state block is similar to Algorithm 3 due to the shared structure of the city and state blocks.

### 4.5. Temporary and Permanent Block Consensus Process

In our approach, we implemented a simple majority vote consensus mechanism for publishing new hospital, city, and state blocks. The consensus processes implemented in HBN, CBN and SBN function similarly. Let $\lambda$ be the total number of super-peer agents from a blockchain network who participate in a consensus process. The block announcer, the super-peer agent who is responsible for initiating the consensus process, must broadcast the new block to other super-peer agents in the network and gather at least $\lambda/2$ approvals from super-peer agents within the same blockchain network.

Figure 8 shows a general illustration of the consensus process for approving a new temporary block in an HBN. From the figure, we can see that the temporary block consensus process consists of 7 steps. The first step is the announcement of a newly created temporary block $TCB_j$ by the block announcer $\beta_{HOS\_A}$ to the super-peer agents of other hospitals in the network. To simplify matters, we show only one such agent in the figure, i.e., $\beta_{HOS\_B}$. Note that hospital regular-peer agent $\beta_{HREP\_C}$ does not participate in the consensus process as it does not have direct access to the CHB. Once the announcement is broadcast and received, $\beta_{HOS\_B}$ retrieves $TCB_j$ from the block announcer in step 2. After that, in step 3, $\beta_{HOS\_B}$ verifies the validity of $TCB_j$ by checking the integrity of $TCB_j$ and the digital signature of the block announcer in the block. If $TCB_j$ is considered valid, $\beta_{HOS\_B}$ creates its digital signature and sends it back to the block announcer as an approval vote in step 4. The block announcer waits for a certain amount of time in step 5 until either a timeout is reached, or a majority of approval votes are collected. All valid digital signatures are added to the digital signature list of $ds[TCB_j]_v$. If a majority vote is received by the block announcer $\beta_{HOS\_A}$, block $TCB_j$ is considered complete and can be added to the CHB. In this case, agent $\beta_{HOS\_A}$ notifies $\beta_{HOS\_B}$ that block $TCB_j$ has successfully passed the consensus process in step 6. Finally, in step 7, each hospital super-peer agent with a completed $TCB_j$ can generate a lite temporary block $TLB_j$ and broadcast it to its respective regular-peer and hospital regular-peer agents for inclusion of $TLB_j$ in their LHBs.

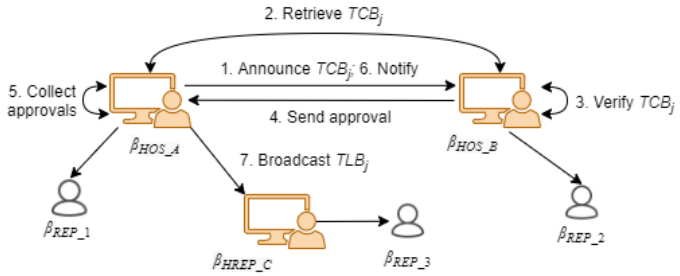


Figure 8: The Consensus Process for Approving a New Temporary Block

As more and more temporary blocks are added to the blockchain by a hospital super-peer agent $\beta_{HOS}$ through the temporary block consensus process, agent $\beta_{HOS}$ can decide to merge all its own published temporary blocks into one permanent block by initiating the permanent block consensus process. Note that when a permanent block consensus process is initiated, no other permanent block or temporary block consensus processes are allowed to occur at the same time and vice versa. The consensus process for approving a new permanent block in an HBN is similar to that for approving a new temporary block in an

HBN, depicted in Figure 8, but it requires the deletion of all temporary blocks that have been merged into a new permanent block in its last step. Finally, the consensus process for approving a new city or state block in a CBN or the SBN, respectively is also similar. More details can be found in a recent work [7].

## 5. The Search and Retrieval Processes for EHRs

There are numerous peers involved in the blockchain networks, playing different roles such as doctors, nurses and patients. Before retrieving EHRs from the blockchains, it is critical to assign appropriate permissions to each role to access the EHRs stored in the blockchains and protect them from unauthorized access [22]. In this section, we first describe the RBAC policies used in our approach, and then present our integrated search and retrieval process for EHRs.

### *5.1. Role-Based Access Control Policy*

In our previous work, we implemented RBAC policies in an HBN as mandatory rules that specify which data in a blockchain can be accessed by participants based on their credentials [5]. With the introduction of hierarchical blockchain framework, RBAC policies are required to function effectively across all three layers of the blockchain networks. In other words, RBAC policies must be defined to grant access to a patient's EHRs across hospitals, cities, or states. These RBAC policies are stored in $HR_{ACP}$, $CR_{ACP}$ and $SR_{ACP}$ of a hospital blockchain, a city blockchain and a state blockchain, respectively. A patient is required to decide whether to allow or deny the sharing of their medical data with other hospitals within the city, state, or country prior to a doctor's visit. Any relevant access control policies are then created and added to the appropriate hospital, city, or state blockchains. A regular peer agent $\beta_{REP}$ that represents an end user (e.g., a doctor), must seek permission from a hospital, city, or state super-peer agent for access to a patient's EHRs stored within hospitals either locally or across the country. An example policy H1 is shown below, which is stored as an $HR_{ACP}$ in a hospital blockchain and enforced by the hospital super-peer agents within the corresponding HBN.

**policy** H1 {
  **summary**: Doctor D#111 from Hospital_1 is allowed access to Patient P#112's EHRs in Hospital_2.
  **hospitals**: Hospital_1; Hospital_2
  **role**: doctor (Doctor D#111), patient (Patient P#112)
  **condition**: doctor ∈ Hospital_1 && patient ∈ Hospital_2
  **owners**: $\beta_{HOS\text{-}HOSPITAL\_1}$; $\beta_{HOS\text{-}HOSPITAL\_2}$
  **expiration**: 01/01/2031
}

Access control policy H1 specifies that doctor D#111 is allowed to access patient P#112's EHRs in Hospital_2. Note that since a hospital access control policy $HR_{ACP}$ specifies access rights within an HBN, we can safely assume that Hospital_1 and Hospital_2 are located in the same city. When doctor D#111 makes a request to access patient P#112's EHRs, both hospital super-peer agents $\beta_{HOS\text{-}Hospital\_1}$ and $\beta_{HOS\text{-}Hospital\_2}$ attempt to verify the request by checking policy H1 stored in their blockchains. If approved, doctor D#111 is granted access to patient P#112's EHRs maintained by Hospital_2. A city access control policy stored as a $CR_{ACP}$ is similar to a hospital access control policy stored as an $HR_{ACP}$, but it must specify the cities where the hospitals are located because the hospitals belong to different cities within the same state; otherwise, if the hospitals belong to the same city, the access control policy shall be recorded as an $HR_{ACP}$. An example policy C1 is shown below, which can be stored in a city blockchain as a $CR_{ACP}$ and enforced by city super-peer agents in CBNs for the HBNs under their jurisdiction.

**policy** C1 {
  **summary**: Doctor D#111 from Hospital_1 (City_1) is allowed access to Patient P#112's EHRs in Hospital_3 (City_3).
  **hospitals**: City_1.Hospital_1; City_3.Hospital_3
  **role**: doctor (Doctor D#111), patient (Patient P#112)
  **condition**: doctor ∈ City_1.Hospital_1 && patient ∈ City_3.Hospital_3
  **owners**: $\beta_{CIT\text{-}City\_1}$; $\beta_{CIT\text{-}City\_3}$
  **expiration**: 02/02/2032
}

Access control policy C1 specifies that doctor D#111 from Hospital_1 in City_1 is allowed to access patient P#112's EHRs at Hospital_3 in City_3. Different from policy H1, when doctor D#111 makes a request to access patient P#112's EHRs located in a different city, both hospital super-peer agents $\beta_{HOS\text{-}Hospital\_1}$ and $\beta_{HOS\text{-}Hospital\_3}$ forward the request to their city super-peer agents $\beta_{CIT\text{-}City\_1}$ and $\beta_{CIT\text{-}City\_3}$ to check against policy C1 stored in their city blockchains. If approved, doctor D#111 is granted access to patient P#112's EHRs at Hospital_3 in City_3.

A state access control policy stored as an $SR_{ACP}$ is similar to a city access control policy stored as a $CR_{ACP}$, but it must specify both the cities and the states where the hospitals are located. an example policy S1 is shown below, which can be stored as an $SR_{ACP}$ in a state blockchain and enforced by state super-peer agents in the SBN for CBNs and HBNs under their jurisdiction.

**policy** S1 {
  **summary**: Doctor D#111 from Hospital_1 (City_1, State_1) is allowed access to Patient P #112's EHRs in Hospital_4 (City_4, State_4).
  **hospitals**: State_1.City_1.Hospital_1; State_4.City_4.Hospital_4
  **role**: doctor (Doctor D#111), patient (Patient P#112)
  **condition**: doctor ∈ State_1.City_1.Hospital_1 && patient ∈ State_4.City_4.Hospital_4
  **owners**: $\beta_{STA\text{-}State\_1}$; $\beta_{STA\text{-}State\_4}$
  **expiration**: 03/03/2033
}

Access control policy S1 specifies that doctor D#111 from Hospital_1 (City_1, State_1) is allowed to access patient P#112's EHRs stored in Hospital_4 (City_4, State_4). In a similar nature to policy C1, when doctor D#111 makes a request to access patient P#112's EHRs located in a different state, both hospital super-peer agents $\beta_{HOS\text{-}Hospital\_1}$ and $\beta_{HOS\text{-}Hospital\_4}$ forward the request to their state super-peer agents $\beta_{STA\text{-}State\_1}$ and $\beta_{STA\text{-}State\_4}$, through their city super-peer agents, $\beta_{CIT\text{-}City\_1}$ and $\beta_{CIT\text{-}City\_4}$. The request is then checked against policy S1 stored in their state blockchains. If approved, doctor D#111 is granted access to patient P#112's EHRs from Hospital_4 (City_4, State_4).

Note that to support efficient access authorization and avoid duplication of an access control policy across multiple access control policy records, access control policies are no longer encrypted as in our previous work [7]. For more examples of access control policies at hospital, city and state levels, refer to earlier work [5], [7].

### *5.2. Integrated Search and Retrieval of EHRs*

Once the required access control policies have been created and stored in the relevant hospital, city and state blockchains, the

associated data can now be opened and shared with other hospitals across the country. This data sharing is supported by an integrated EHRs search and retrieval process that enables those with the proper authorization to retrieve all EHRs of a patient from any hospitals, regardless of which HBNs they participate in. This process involves all three layers of our hierarchical blockchain framework, as search requests are forwarded and executed concurrently across all super-peer agents in the hierarchical network structure. The concurrent search and retrieval process is defined by three procedures, which are searching and retrieving EHRs across hospitals within the same city, searching and retrieving EHRs across cities within the same state, and searching and retrieving EHRs across states within a country. We now describe each of the three procedures as follows.

The procedure of searching and retrieving EHRs across hospitals within the same city is presented in Algorithm 4. The algorithm is initiated by a hospital super-peer agent $\beta_{HOS}$ on behalf of its end user (e.g., a doctor) to search and retrieve patient $p$'s EHRs from other hospitals within the same city (i.e., within the same HBN). Agent $\beta_{HOS}$ sends this request to its city super-peer agent $\beta_{CIT}$ to start the process.

**Algorithm 4: Searching and Retrieving a Patient's EHRs from All Hospitals within the Same City by a City Super-Peer Agent $\beta_{CIT}$**

**Input:** A retrieval request for hospitals containing patient $p$'s EHRs
**Output:** A list of links to patient $p$'s EHRs

1. Let $\rho_{h\_list}$ be the list of hospital super-peers under $\beta_{CIT}$'s jurisdiction
2. Let $\eta_{ehr\_hlist}$ be an empty list of links to EHRs; *nResponse* = 0
3. **for** each $\gamma_h$ in $\rho_{h\_list}$
4. forward the retrieval request to $\gamma_h$ asynchronously, which invokes a search process at hospital $h$ based on the established policies
5. **while** (*not* timeout) or *nResponse* != | $\rho_{h\_list}$ |
6. **if** $\gamma_h$ returns a link to $p$'s EHRs
7. add the link to the list $\eta_{ehr\_hlist}$; *nResponse*++
8. **else** *nResponse*++; **continue** // $\gamma_h$ returns no link to EHRs
9. **return** the list $\eta_{ehr\_hlist}$

According to the algorithm, agent $\beta_{CIT}$ sends concurrent requests in its HBN to all hospital super-peer agents under its jurisdiction and waits for responses or until the timeout. Each hospital super-peer agent $\gamma_h$ who receives this request will perform a permission checking based on the established access control policies stored as $HR_{ACP}$ in its hospital blockchain. If valid, $\gamma_h$ creates a link that allows access to patient $p$'s EHRs and sends it back to $\beta_{CIT}$. If $\beta_{CIT}$ receives a response from $\gamma_h$ with this link, the link is added to list $\eta_{erh\_hlist}$; otherwise, $\beta_{CIT}$ continues to wait. When all hospital super-peer agents have responded or timed out, the list $\eta_{erh\_hlist}$ is returned and sent back to $\beta_{HOS}$. Upon receiving $\eta_{erh\_hlist}$, $\beta_{HOS}$ can then use the links to access and retrieve patient $p$'s EHRs on behalf of the end user.

The procedure of searching and retrieving EHRs across cities within the same state, is presented in Algorithm 5. Similar to the procedure of searching and retrieving EHRs across hospitals within the same city, Algorithm 5 is initiated by a hospital super-peer agent $\beta_{HOS}$ on behalf of its end user (e.g., a doctor) to search and retrieve patient $p$'s EHRs from hospitals in different cities within the same state (i.e., within different HBNs connected under the same CBN). Agent $\beta_{HOS}$ sends this request to its city super-peer agent $\beta_{CIT}$, who forwards it to its state super-peer agent $\beta_{STA}$ to start the process. According to the algorithm, agent $\beta_{STA}$ sends concurrent requests to all city super-peer agents under its jurisdiction in its CBN and waits for responses from them or until it times out. Upon receiving the search request, each city super-peer agent $\gamma_c$ performs a permission check based on the access control policies stored as $CR_{ACP}$ in its city blockchain. If valid, $\gamma_c$ executes Algorithm 4 to forward the search and retrieval requests to the hospital super-peer agents under its jurisdiction. If $\gamma_c$ returns a list $\eta_{ehr\_hlist}$ containing links to $p$'s EHRs, $\eta_{ehr\_hlist}$ is appended to the list $\eta_{ehr\_clist}$; otherwise, $\beta_{STA}$ continues to wait. When all city super-peer agents have responded or it times out, the list $\eta_{ehr\_clist}$ is returned and sent back to $\beta_{CIT}$, who further sends it back to $\beta_{HOS}$. Upon receiving $\eta_{erh\_clist}$, $\beta_{HOS}$ can then use the links to access and retrieve patient $p$'s EHRs on behalf of the end user.

**Algorithm 5: Searching and Retrieving a Patient's EHRs from All Hospitals within the Same State by a State Super-Peer Agent $\beta_{STA}$**

**Input:** A retrieval request for hospitals containing patient $p$'s EHRs
**Output:** A list of links that can be used to access patient $p$'s EHRs

1. Let $\rho_{c\_list}$ be the list of city super peers under $\beta_{STA}$'s jurisdiction
2. Let $\eta_{ehr\_clist}$ be an empty list of links to EHRs; *nResponse* = 0
3. **for** each $\gamma_c$ in $\rho_{c\_list}$
4. forward the retrieval request to $\gamma_c$ asynchronously, which invokes *Algorithm* 4 to search in city $c$ based on the established policies
5. **while** (*not* timeout) or *nResponse* != | $\rho_{c\_list}$ |
6. **if** $\gamma_c$ returns $\eta_{ehr\_hlist}$ that contains links to $p$'s EHRs
7. append $\eta_{ehr\_hlist}$ to $\eta_{ehr\_clist}$; *nResponse*++
8. **else** *nResponse*++; **continue** // $\gamma_c$ returns an empty list
9. **return** the list $\eta_{ehr\_clist}$

Finally, the procedure of searching and retrieving EHRs across states within a country is similar to Algorithm 5, where the retrieval request is sent from a hospital super-peer agent $\beta_{HOS}$ to its state super-peer agent $\beta_{STA}$. Agent $\beta_{STA}$ then initiates the concurrent searches by broadcasting the request to all other state super-peer agents. Each state super-peer agent $\gamma_s$, representing state $S$, performs a permission checking based on the access control policies stored as $SR_{ACP}$ in the state blockchain. If valid, $\gamma_s$ executes Algorithm 5 to search and retrieve EHRs of patient $p$ from all cities within state $S$. The return result $\eta_{erh\_clist}$ is appended to $\eta_{erh\_slist}$ if it is not empty. When all state super-peer agents have either responded or timed out, the list $\eta_{erh\_slist}$ is returned and sent back to $\beta_{HOS}$ via $\beta_{CIT}$. $\beta_{HOS}$ can then use the links to access and retrieve patient $p$'s EHRs on behalf of the end user.

## 6. Case Study

To demonstrate the feasibility and efficiency of our proposed approach, we conducted experiments and evaluated the performance of our hierarchical approach based on the settings and results of each simulation. In our experimental environment, we utilized multiple servers and computers connected under the same network. The specifications of the servers include Intel® Core™ i7-4790k CPU @ 3.60GHz (4 CPU Cores); 16 GB RAM, Windows 10 OS (64-bit, x64-based processor); and 256 SSD Hard Drive. Our experimental environment also had a recorded Internet speed of 600 Mbps.

### *6.1. Numbers of Published Blocks During a Week*

In the first case study, we test our temporary and permanent block approach by conducting simulations to evaluate the need to

use temporary blocks. We simulate and analyze the number of permanent blocks that can typically be created each day of a week based on predefined threshold values. These threshold values are the maximum total size of 2GB for all accumulated temporary blocks and a maximum of 100 new blocks added during a single day. If neither of the thresholds is reached, a permanent block is always created at the end of the day. The number of temporary blocks added during a day is determined by the number of patient visits. We assume that a patient visit always results in the generation of an EHR, which is saved as an $HR_{MER}$ and immediately published to the blockchain as a temporary block. To simplify our experiments, we focus only on $HR_{MER}$ rather than other record types, i.e., $HR_{AR}$, $HR_{UPR}$, and $HR_{ACP}$, because in real-world scenarios, $HR_{MER}$ is the main contributor of block content in hospital blockchains. For the content or nature of the EHRs stored in each $HR_{MER}$, we use the following experimental settings. Each $HR_{MER}$ includes text-based reports in the size range of [5, 10] KB, while there is also less than a 10% probability of including multimedia files in the size range of [10, 500] MB. Thus, an $HR_{MER}$ must contain text-based report along with possible multimedia files of different sizes. The range of patient visits are based on hospital sizes, where we simulate three different sizes of hospitals. The first type of hospitals has daily patient visits of [10, 100] and is categorized as a small hospital. The second type of hospitals has daily patient visits of [50, 500] and is categorized as a medium hospital. The third type of hospitals has daily patient visits of [100, 1000] and is categorized as a large hospital. Table 1 shows the number of patient visits for each day of a week at each simulated hospital.

Table 1: Numbers of Patient Visits per Day

| Hospital Size | Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|---|
| Large | 900 | 700 | 500 | 550 | 700 | 750 | 850 |
| Medium | 450 | 300 | 150 | 200 | 300 | 350 | 400 |
| Small | 100 | 70 | 50 | 55 | 60 | 75 | 90 |

We now conduct experiments to generate permanent blocks based on the number of patient visits per day. Figure 9 shows the average number of permanent blocks that can be formed at each hospital based on the experimental settings and the numbers of daily patient visits listed in Table 1. As we can see from the figure, even for a large hospital, the number of permanent blocks published per day is limited. The time interval between each addition of permanent blocks can be several hours or even longer, depending on the number of permanent blocks added that day. This indicates a critical need to use temporary blocks to publish data to the blockchain in a timely manner for immediate access without delay. Based on the results in Figure 9, we conclude that medium and small hospitals would benefit the most from our temporary and permanent block approach, as they generate the fewest average numbers of permanent blocks. Figure 10 shows the relationship between the number of permanent blocks formed vs. the number of temporary blocks added during a day at a medium-sized hospital. Since each patient visit results in a new temporary block being created, the number of newly added temporary blocks is equal to the number of new patient visits. According to the simulation results, in medium-sized hospitals, when the number of patient visits increases, the number of new permanent blocks also increases. When the maximum number of patient visits is reached, i.e., 500, the number of new permanent blocks formed daily is between 4 and 7, which is considered to be very acceptable in terms of spatial efficiency.

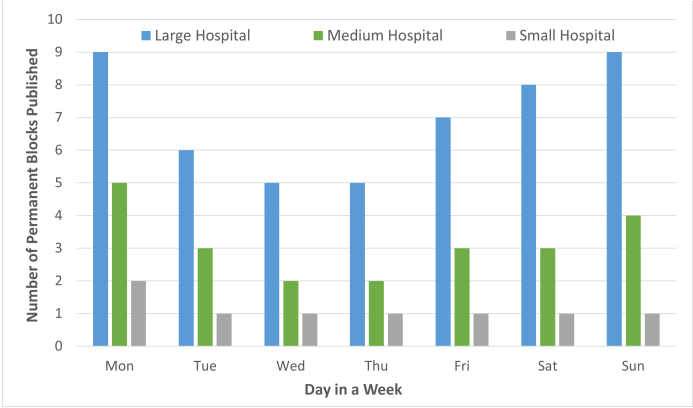


Figure 9: Average Number of Permanent Blocks Published During a Week

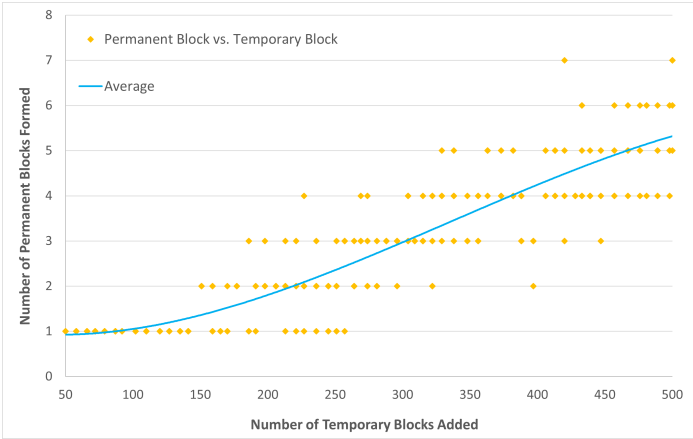


Figure 10: Number of Permanent Blocks Formed vs. Number of Temporary Blocks Added During a Day in a Simulated Medium-Sized Hospital

### 6.2. Latency of Publishing Temporary and Permanent Blocks

In this experiment, we simulate the creation and addition of permanent and temporary blocks to the blockchain. We record and analyze the time to create, broadcast, and publish temporary and permanent blocks based on a varying number of hospital super-peer agents within an HBN. We use the same settings established in the first case study for this experiment. This means that in a typical day, a new permanent block will have either 100 records stored or 2 GB in size, according to the given thresholds. Each temporary block includes only $HR_{MER}$ containing text-based reports and potentially (10%) multimedia files of [10, 500] MB in size. In addition, we add random delays in the range of [100, 3000] milliseconds to simulate network congestion during the consensus process. Figure 11 shows the experimental results and the efficiency of our approach using temporary blocks. From the figure, we can see that at most, it takes about less than half a minute to create a temporary block and add it to the blockchain. Due to the large variation in the potential size and frequency of temporary blocks containing multimedia files in our experimental settings, the range between each case can vary considerably. In contrast, for permanent blocks, the range is more consistent for each case due to previously determined size and number thresholds. In general, the overall time for a permanent block to

be created and added to the blockchain is less than one minute. While there is no big data transferred during the consensus process, additional validation is required by the other super-peer agents to verify the permanent block broadcast by the block announcer and the associated temporary blocks previously stored in their local copies of the blockchain. This significantly increases the overall time required for the consensus process, which takes longer time when compared to the publication process involving temporary blocks only. Nevertheless, it takes no more than 20 seconds to create and add a temporary block to the blockchain, which allows many consensus processes for new temporary blocks to be performed during a single day without encountering significant delays. Therefore, we can conclude that our approach supports efficient creation and addition of temporary and permanent blocks to the blockchain.

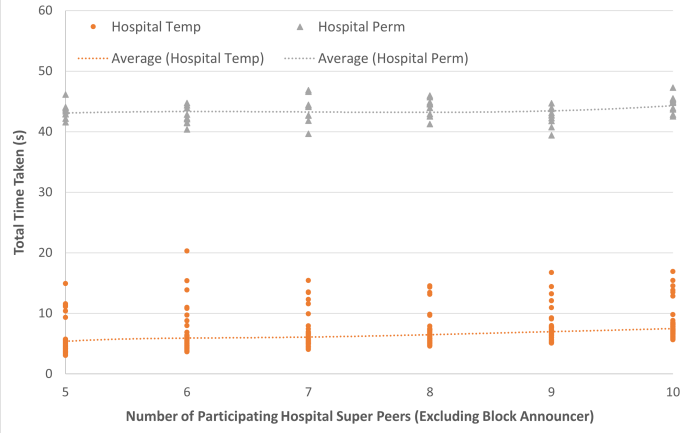


Figure 11: Total Time Taken to Create, Broadcast, and Publish Hospital Temporary and Permanent Blocks

### *6.3. Search and Retrieval Time in Hierarchical Blockchains*

In this experiment, we simulate the integrated concurrent search and retrieval processes for patient EHRs. We record and analyze the total time taken to search and retrieve a number of patient's EHRs in the hierarchical blockchain networks. Our experimental environment consists of three layers of fully simulated hierarchical networks with a varying number of HBNs, CBNs, and an SBN. We also have a range of [6, 10] hospital super peers within an HBN, [100, 500] city super peers within a CBN, and 50 state super peers within the SBN. The contents of our hospital, city and state blockchains contain all the necessary or relevant access control policy records to enable our searching process. To simplify the overall downloading process, the hospital blockchain contains only EHRs with multimedia files. Each EHR has a range of [10, 500] MB in size, similar to our previous case studies. The EHRs are stored in different hospital blockchains within different HBNs to simulate a patient who visits multiple hospitals in different cities. We also introduce a random delay with a range of [100, 3000] milliseconds to simulate network congestions. In addition, we assume that all hospitals have the required infrastructure to allow multiple concurrent file downloads to mitigate throttling when any number of peers download multiple files simultaneously. Each hospital agent also maintains a separate local index file for efficient responses to any EHR-related inquiries. Figure 12 shows the total time taken to search and retrieve different numbers of EHRs. Based on the figure, we can see that the search time remains relatively constant regardless of the number of EHRs to be searched. Several factors, such as the use of separate index files to track patient EHRs for fast response and the small size of the metadata involved during the concurrent search process, contribute significantly to this stability. However, when the total numbers of EHRs to be retrieved increases, the time to retrieve those EHRs also increases. This result is consistent with the experimental results we reported in our previous work [7]. However, the current approach is more efficient compared to the previous method as the waiting time for creating new access policy records is not needed any more after the search process. This leads to an overall time improvement, which allows multiple EHRs to be searched and retrieved in one integrated process.

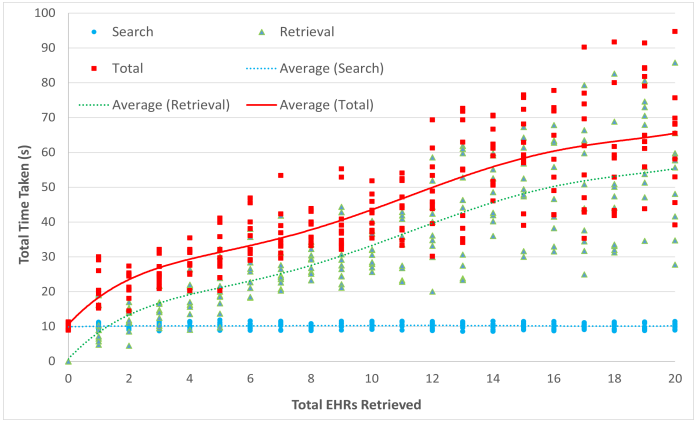


Figure 12: Time Taken to Search and Retrieve Varying Number of EHRs

## 7. Conclusions and Future Work

In this paper, we build on the concepts and methods of previous work [7] and further explore them by introducing several design changes. These changes include limiting the number of hospital super-peer agents in an HBN, adopting a temporary and permanent block scheme, and integrating the search and retrieval processes for EHRs. The number of hospital super-peer agents in an HBN is limited to minimize the redundancy of big data stored in the blockchain. This also ensures better scalability, as we limit the growth potential of hospital blockchain to a more manageable level. The use of the temporary and permanent block scheme in our hierarchical blockchain approach ensures timely publications of EHRs in an HBN. Any urgent data can be published in a temporary block immediately, while once a certain threshold has been reached, a permanent block consisting of a number of temporary blocks can be formed. This scheme allows for timely and space-saving publications of EHRs to the cloud-based hospital blockchains. Finally, the search and retrieval processes for EHRs have been integrated to be more efficient. As the experimental results show, our new hierarchical blockchain approach is efficient and effective, allowing for a timely and spatially efficient publication of EHRs to the hospital blockchain as well as a better overall performance in the integrated search and retrieval process of EHRs.

In future work, we plan to perform an in-depth comparison of our cloud-based on-chain blockchain approach with IPFS-based off-chain approaches [10], [11] and mechanisms for reliable and secure distributed cloud data storage [23]. We will focus on the

redundancy and efficiency aspects of such comparisons and evaluate the performance of our cloud-based hierarchical blockchain mechanism. In addition, we plan to further improve and develop our approach to defend against real-world attacks, such as DDOS attacks and insider threats [24], [25], [26]. An emphasis will be on evaluating the performance of our consensus process and cryptographic procedures against potential attacks and improving them if necessary.

## Conflict of Interest

The authors declare no conflict of interest.

## References


[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," October 2008. Retrieved on January 15, 2021 from https://bitcoin.org/bitcoin.pdf.

[2] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, E. B. Hamida, "Consortium blockchains: overview, applications and challenges," International Journal on Advances in Telecommunications, **11**(1&2), 51-64, 2018.

[3] M. T. de Oliveira, L. H. A. Reis, R. C. Carrano, F. L. Seixas, D. C. M. Saade, C. V. Albuquerque, N. C. Fernandes, S. D. Olabarriaga, D. S. V. Medeiros, D. M. F. Mattos, "Towards a blockchain-based secure electronic medical record for healthcare applications," in Proceedings of the 2019 IEEE International Conference on Communications (ICC), 1-6, Shanghai, China, May 2019, doi: 10.1109/ICC.2019.8761307.

[4] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, M. Guizani, "MeDShare: trust-less medical data sharing among cloud service providers via blockchain," *IEEE Access*, **5**, 14757-14767, July 2017, doi: 10.1109/ACCESS.2017.2730843.

[5] A. Thamrin, H. Xu, "Cloud-based blockchains for secure and reliable big data storage service in healthcare systems," in Proceedings of the 15th IEEE International Conference on Service-Oriented System Engineering (IEEE SOSE 2021), 81-89, Oxford Brookes University, UK, August 2021, doi: 10.1109/SOSE52839.2021.00015.

[6] R. Ming, H. Xu, "Timely publication of transaction records in a private blockchain," 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), 116-123, Macau, China, December 2020, doi: 10.1109/QRS-C51114.2020.00030.

[7] A. Thamrin, H. Xu, "Hierarchical cloud-based consortium blockchains for healthcare data storage," in 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C), 644-651, Hainan Island, China, December 2021, doi: 10.1109/QRS-C55045.2021.00098.

[8] S. Alexaki, G. Alexandris, V. Katos, N. E. Petroulakis, "Blockchain-based electronic patient records for regulated circular healthcare jurisdictions," in Proceedings of the 23rd IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 1-6, 2018, doi: 10.1109/CAMAD.2018.8514954.

[9] R. Kumar, N. Marchang, R. Tripathi, "Distributed off-chain storage of patient diagnostic reports in healthcare system using IPFS and blockchain," in Proceedings of the 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), 1-5, Bengaluru, India, 2020, doi: 10.1109/COMSNETS48256.2020.9027313.

[10] D. Li, W. E. Wong, M. Zhao, Q. Hou, "Secure storage and access for task-scheduling schemes on consortium blockchain and interplanetary file system," 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), 153-159, 2020, doi: 10.1109/QRS-C51114.2020.00035.

[11] Y. Jeong, D. Hwang, K. Kim, "Blockchain-based management of video surveillance systems," in Proceedings of the 2019 International Conference on Information Networking (ICOIN), 465-468, Kuala Lumpur, Malaysia, 2019, doi: 10.1109/ICOIN.2019.8718126.

[12] Z. Su, H. Wang, H. Wang, X. Shi, "A financial data security sharing solution based on blockchain technology and proxy re-encryption technology," in Proceeedings of the IEEE 3rd International Conference of Safe Production and Informatization (IICSPI), 462-465, Chongqing City, China, 2020, doi: 10.1109/IICSPI51290.2020.9332363.

[13] H. Wang, Y. Song, "Secure cloud-based EHR system using attribute-based cryptosystem and blockchain," Journal of Medical Systems, **42**(152), 1-9, July 2018, doi: 10.1007/s10916-018-0994-6.

[14] B. S. Egala, A. K. Pradhan, V. R. Badarla, S. P. Mohanty, "Fortified-chain: a blockchain based framework for security and privacy assured Internet of medical things with effective access control," IEEE Internet of Things Journal, **8**(14), 11717-11731, July 2021, doi: 10.1109/JIOT. 2021.3058946.

[15] A. Fernandes, V. Rocha, A. F. d. Conceicao, F. Horita, "Scalable architecture for sharing EHR using the Hyperledger blockchain," in Proceedings of the IEEE International Conference on Software Architecture Companion (ICSA-C), 130-138, Salvador, Brazil, March 2020, doi: 10.1109/ICSA-C50368.2020.00032.

[16] N. Nicol, H. Xu, "A blockchainless approach for trusted public construction bidding," Computer and Information Science Technical Report, Computer and Information Science Department, University of Massachusetts Dartmouth, December 2018.

[17] L. Cui, S. Yang, Z. Chen, Y. Pan, M. Xu, K. Xu, "An efficient and compacted DAG-based blockchain protocol for industrial Internet of things," IEEE Transactions on Industrial Informatics, **16**(6), 4134-4145, 2020, doi: 10.1109/TII.2019.2931157.

[18] A. Buzachis, A. Celesti, M. Fazio, M. Villari, "On the design of a blockchain-as-a-service-based health information exchange (BaaS-HIE) system for patient monitoring," in Proceedings of the IEEE Symposium on Computers and Communications (ISCC), 1-6, Barcelona, Spain, July 2019, doi: 10.1109/ISCC47284.2019.8969718.

[19] D. C. Nguyen, P. N. Pathirana, M. Ding, A. Seneviratne, "Blockchain for secure EHRs sharing of mobile cloud based e-health systems," IEEE Access, **7**, 66792-66806, May 2019, doi: 10.1109/ACCESS. 2019.2917555.

[20] H. Guo, W. Li, M. Nejad, C. Shen, "Access control for electronic health records with hybrid blockchain-edge architecture," in Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain), 44-51, July 2019, doi: 10.1109/Blockchain.2019.00015.

[21] H. Guo, W. Li, E. Meamari, C. Shen, M. Nejad, "Attribute-based multi-signature and encryption for EHR management: a blockchain-based solution," in Proceedings of the 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 1-5, Toronto, ON, Canada, May 2020, doi: 10.1109/ICBC48266.2020.9169395.

[22] M. Meingast, T. Roosta, S. Sastry, "Security and privacy issues with health care information technology," in Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society, 5453-5458, New York, NY, USA, September 2006, doi: 10.1109/IEMBS.2006.260060.

[23] H. Xu, D. Bhalerao, "Reliable and secure distributed cloud data storage using Reed-Solomon codes," International Journal of Software Engineering and Knowledge Engineering (IJSEKE), **25**(9&10), 1611-1632, 2015, doi: 10.1142/S0218194015400355.

[24] S. Northcutt, J. Novak, Network intrusion detection, 3rd Edition, Sams Publishing, August 2002.

[25] H. Xu, A. Reddyreddy, D. F. Fitch, "Defending against XML-based attacks using state-based XML firewall," Journal of Computers (JCP), **6**(11), 2395-2407, November 2011, doi: 10.4304/jcp.6.11.2395-2407.

[26] J. Mirkovic, P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," **34**(2), 39-53, April 2004, doi: 10.1145/997150.997156.