Cloud-Based Blockchains for Secure and Reliable Big Data Storage Service in Healthcare Systems

Alvin Thamrin
Computer and Information Science Department
University of Massachusetts Dartmouth
Dartmouth, MA 02747, USA
athamrin@umassd.edu

Haiping Xu
Computer and Information Science Department
University of Massachusetts Dartmouth
Dartmouth, MA 02747, USA
hxu@umassd.edu

***Abstract*—Due to regulations and policies enacted to protect patients' privacy, electronic health records (EHRs) must be kept as sensitive information in a secure and reliable manner. Maintaining EHRs by hospitals and safely sharing them with others have proved to be challenging tasks. In this paper, we introduce a cloud-based blockchain (CBC) approach to achieving data accessibility, redundancy, and security for storing and sharing EHRs. A lightweight CBC, called lite blockchain (LBC), is designed to store EHRs' metadata and text-based information locally. Our approach allows for big data to be safely stored in CBCs, and for information to be efficiently retrieved via LBCs. To restrict and grant access of the data to authorized participants only, we present our data security mechanism along with role-based access control policies. The experimental results show that our cloud-based blockchain approach is feasible and efficient for accessing and sharing EHRs stored both locally and in clouds.**

Keywords—Cloud-based blockchain, lite blockchain, electronic health record (EHR), multimedia files, role-based access control

I. Introduction

Electronic health records (EHRs) are defined as information that is associated with regular patient data or as part of a clinical trial program stored electronically in a digital format. These records contain numerical data (e.g., heart rate and temperature), diagnostic-related information (e.g., blood tests and genetic tests), imagery (e.g., X-rays and CT scan), prescription data and more. Such data are considered confidential, and thus must be kept in a secure fashion. Accessibility must also be considered when storing these data to avoid complications on future access and data retrieval by authorized parties.

The blockchain technology provides a decentralized ledger or storage scheme, where data can be stored and shared easily among participants in a blockchain network. A blockchain consists of a growing list of records, called blocks, which are linked using cryptography and can record transactions between two parties efficiently and in a verifiable and permanent way [1]. The blockchain technology can be adopted in healthcare systems, where successful implementations ensure preservation of medical history of patients, and would also allow doctors and patients to easily share important medical data with each other even in a cross-hospital setting [2]. Storing medical data in blockchains, however, presents several problems related to growth sustainability and data privacy.

A blockchain ledger generally only grows with data being added to it and considered to be immutable. Reversing this process to remove data in the blockchain is very difficult and costly to all participants. Unfortunately, EHRs are typically considered to be big data as some of them are recorded as multimedia files including X-ray, CT scan, MRI, and ultrasound videos. Placing such data into blockchains would generate bulky block files, and disseminating these blocks to all participants in the blockchain network is nontrivial due to the heavy strain on participants' bandwidth as well as memory and storage space on their local machines. Thus, a common approach to this dilemma usually involves the usage of an off-chain storage to store big data, while using the blockchain to store metadata that is significantly smaller in size to save space [3-4]. A more challenging approach is to embrace the idea of putting big data into the blockchain, so we can take advantage of the benefits of using the blockchain technology such as immutability, redundancy, and tamper resistance. However, issues related to bulky blocks must be handled to ensure ease of synchronization among participants in a blockchain network.

In this paper, we introduce a cloud-based blockchain (CBC) approach along with a lightweight CBC, called lite blockchain (LBC), to allow the storage of big data in blockchains for healthcare systems. A CBC stores all EHRs including multimedia files in the clouds, thus local memory and storage space investment would not prove to be much of a challenge. On the other hand, an LBC stores the same data as in a CBC except the multimedia files. To make the multimedia files searchable, their metadata are kept in the LBC. All regular peers of a blockchain network incorporate the LBC locally, leading to a better ease of synchronization and space upkeep on their local machines. A regular peer can use the metadata stored in the LBC to request the corresponding multimedia files stored in the CBC via a super peer agent representing a healthcare organization. Our approach involves multiple super peer agents, each of which has a private cloud maintained by a cloud manager agent. This results in multiple synchronized copies of the CBC stored in multiple private clouds, leading to increased redundancy and resilience against scenarios such as tampering and loss of data that are ever present when data is stored in clouds [5].

Another challenge in this research is to ensure data security. Since EHRs are sensitive information, it can only be accessed by those with proper credentials. Therefore, storing EHRs in the blockchain without any form of access control and encryption is forbidden, as it violates patient privacy laws such as HIPAA. In response to this concern, we utilize public and private key schemes, alongside role-based access control (RBAC) policies to restrict access to data stored in the blockchains either locally or in the clouds. In this manner, we are able to achieve accessibility, redundancy, and security for all EHRs stored in blockchain-based healthcare systems.

2642-6587/21/$31.00 ©2021 IEEE
DOI 10.1109/SOSE52839.2021.00015

II. Related Work

The blockchain technology is one of the contemporary innovations that have been growing in popularity in many applications including secure storage of EHRs. In the healthcare domain, there has been a pressing need to have an effective framework to store and share these medical records online. Oliveira et al. proposed a blockchain-based approach to storing and sharing electronic medical records [6]. In their approach, medical records are encrypted before they are stored in the blockchain and the keys are shared between the patients and their trusted healthcare workers. Alexaki et al. described a similar approach to using blockchain to enable the sharing of medical data between healthcare providers, while respecting the patient privacy and preserving the data's integrity [7]. They presented a conceptual medical record access and sharing mechanism and showed that their approach was suitable for regulated healthcare jurisdictions. Zhao et al. proposed a key management scheme using a body sensor network to secure data in a health blockchain [8]. They utilized biosensor nodes to collect physiological data for generating and recovering the keys used to encrypt and decrypt medical data stored in the blockchain. Rui and Xu recently introduced a novel blockchain framework to publish time-sensitive healthcare information such as COVID-19 cases and testing records [9]. They used temporary blocks to facilitate timely publication of new medical records, and then combined them into a permanently published block. Although the above methods addressed the security concerns or efficiency issues in storing medical data in blockchains, they did not provide the needed solutions to storing big data in blockchains in a reliable and secure manner. For example, the work presented in [6] mentioned storing only the hash values of large medical records into the blockchain without detailing on how to effectively store the data themselves. It should be noted that storing big data in blockchains would cause maintenance and scalability issues due to the storage of numerous large medical records. In our approach, we use CBCs to store large medical records in private clouds, while utilizing its lightweight counterparts, the LBCs, to maintain and store text-based information and metadata of the medical records.

There have been studies that address the challenges of using blockchain approaches to storing big data including images and videos. However, those studies generally utilize off-chain methods to store big data instead of on-chain solutions. Kumar and Tripathi proposed an InterPlanetary File System (IPFS) based storage model to store large transactions using their hash values to reduce the size of each block in a blockchain [10]. The complete transactions that contain image files, are stored in the IPFS, while the blocks in the blockchain only include the returned IPFS hash of the transactions to save space. This work is further expanded in [4], where the authors proposed a similar approach using blockchain and IPFS to store and share patient diagnostic reports. In their design, actual patient reports are stored in a distributed off-chain storage using IPFS, while the blockchain only stores hashes of reports. Wang and Song proposed a secure EHR system using attribute-based cryptosystem and blockchain technology to secure medical data stored in the cloud [11]. Similar to other off-chain approaches, large-scale medical data are stored in the cloud and the blockchain only stores the metadata of EHRs. Different from the above methods, our approach stores all EHRs, with the inclusion of multimedia files, in CBCs. The LBCs, which contain the metadata of these EHRs, can be used to search for all EHRs including the multimedia files stored in CBCs. Thus, unlike the existing off-chain approaches, our approach preserves the benefits of data immutability, integrity, and availability that come with the blockchain structure for all medical data.

Additionally, there are studies that discuss how to use the access control mechanism in a blockchain framework to restrict access of sensitive information to authorized parties only. Guo et al. proposed a hybrid architecture of using both blockchain and edge nodes to facilitate attribute-based access control of EHR data [12]. They used smart contracts provided by the Hyperledger Composer Fabric framework to enforce access control of EHR data stored in the off-chain edge nodes. Their work has been further developed to make up the lack of encryption for data stored in the edge nodes [13]. The authors utilized multi-authority attribute-based encryption (ABE) scheme to encrypt EHR data stored in the edge nodes and attribute-based multi-signature (ABMS) scheme to authenticate user signatures that are integrated with the smart contracts used to enforce the access control policies. Nguyen et al. utilized a smart contract on an Ethereum blockchain to design an access control mechanism on managing user access to ensure efficient and secure EHRs sharing on mobile devices [14]. Similar to the above approaches, the access control mechanism adopted in the proposed design is too general and broad to handle different medical providers. This proved to be problematic as hospitals may have their own access control policies to follow. In contrast, our approach specifies hospital-wide RBAC policies, restricting the access permissions of users registered with different healthcare providers, on information stored in the blockchains. The RBAC polices are enforced by multiple super peer agents and cloud manager agents, who represent different healthcare organizations. Note that in our approach, the RBAC polices are stored in the blockchain and can be updated by storing new policies in a new block of the blockchain. Thus, our RBAC-based approach is a more comprehensive and reliable one that enforces access control based on different hospital-wide policies stored and accessed in the blockchains.

III. A Framework for Cloud-based Blockchains

Let the length of a CBC and its lightweight version LBC be h. A cloud-based block CB_i and a lite block LB_i, where $1 \leq i \leq h$, contain identical core healthcare information such as medical data, user information, and access control policies. The main difference between them is the exclusion of multimedia files, such as MRI scan image and ultrasound video, in LB_i for the purpose of saving storage space on local machines. This allows end users to retrieve any contents stored in a CBC via a LBC on their local machines.

The participants of the blockchain network include regular peer agents β_{REPs}, super peer agents β_{SUPs}, and cloud manager agents β_{CLMs}. In the context of the healthcare domain, β_{REPs} are software agents representing doctors, nurses, and patients, who are end users of the blockchain network. Participants β_{SUPs} are defined as software agents with the authority to create and vote on a new block. An agent β_{SUP} represents a hospital, and only one β_{SUP} per hospital is allowed in the network. A participant β_{CLM} is a software agent, who maintains a private cloud owned by a hospital represented by a β_{SUP}. Agents β_{REPs} and β_{SUPs} are permitted to have a copy of the LBC, while each β_{CLM} manages a copy of the CBC. Fig. 1 shows an example of their relationships in a blockchain network. In the figure, hospital A is represented by super peer agent β_{SUP_A}, who communicates with regular peer agents β_{REP_1} and β_{REP_2}, representing doctors, nurses or patients employed or enrolled in hospital A. Software agents β_{REP_1}, β_{REP_2} and β_{SUP_A} are also connected to their hospital's private cloud, i.e., *PrivateCloud_A*, managed by cloud manager agent β_{CLM_A}. Agent β_{CLM_A} enables access to any EHRs, with proper permission, that are stored in its CBC. Agents β_{SUPs}, representing different hospitals, can directly communicate with each other in a consensus process to decide whether a new block can be added to the blockchain. During the consensus process, a new cloud-based block CB_{h+1} containing selected records needs to gain a majority approval from the agents β_{SUPs}. If CB_{h+1} is approved, the new block is broadcast to all β_{CLMs} to be added to their CBCs. Meanwhile, a lite block LB_{h+1} derived from CB_{h+1} is broadcast to all β_{REPs} and β_{SUPs} to be added to their LBCs.

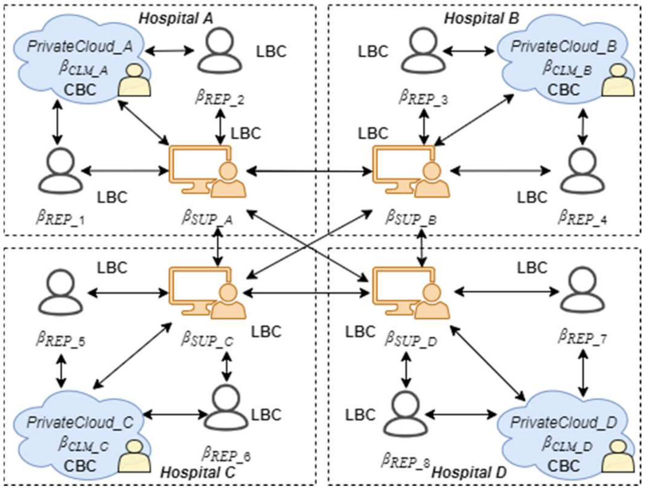

Fig. 1. Relationship between participants in the blockchain network

In our approach, regular peer agents β_{REPs} communicate with their respective β_{SUP} and β_{CLM} to access data stored in the blockchains as well as to submit new records. An agent β_{REP} is required to get permission from its β_{SUP} to access a patient's information. The β_{SUP} will either approve or reject said permission based on the established access control policies. If approved, β_{REP} is granted access to view and retrieve any stored EHRs regarding said patient. If the information in question is text-based, β_{REP} can retrieve it directly from its LBC. Otherwise, if the information includes multimedia files, β_{SUP} sends the metadata of the multimedia files to β_{CLM}, then β_{CLM} extracts the files from its CBC and sends β_{REP} the links to download the files. Note that any sensitive data, including EHRs, stored in the blockchains are in ciphertext form and can only be accessed by authorized parties. In addition, while not shown in Fig. 1, β_{REPs} can also communicate with each other to synchronize their LBCs via the peer-to-peer (P2P) network.

As examples of some general rules, patients are only allowed to access their own EHRs, while doctors can access multiple patients' EHRs within the same hospital. Super peer agents β_{SUPs} implement and enforce access control policies, which dictate the data that each regular peer is allowed to access. An agent β_{SUP} also manages EHRs submitted by β_{REPs} and places them into a new block. The new block then undergoes a consensus process to determine whether it can be added to the blockchain. Data stored in the blockchain are considered immutable and serve as permanent records for patients, who may registered with one or more hospitals in the blockchain network.

IV. Block Structure in a Cloud-Based Blockchain

In this section, we define the structure of a block in a CBC. Since an LBC is a lightweight CBC, a lite block shares the similar structure of a cloud-based block.

A. Block Record Types

There are three different types of block records that can be stored in a CBC or LBC block, namely a user profile R_{UPR}, a set of access control policies R_{ACP}, and a medical record R_{MER} resulting from a doctor's visit. An R_{UPR} contains the user account information of a regular peer agent β_{REP} that participates in the network, which is defined as a 6-tuple (ID, NM, PRK, PUK, SEK, TS), where ID is a unique user identification in the blockchain network; NM is the full name of the end user represented by β_{REP}; PRK and PEK are a pair of β_{REP}'s asymmetric keys; SEK is a symmetric key of β_{REP}; and TS is the time when the record is created. The key SEK is stored in ciphertext $E(SEK, PUK_{SUP})$, encrypted using the public key PUK_{SUP} of agent β_{SUP}, who represents the hospital that β_{REP} has registered with. An R_{UPR} is created when a new participant joins the network or when an existing user profile is updated. Therefore, an agent β_{REP} may have multiple R_{UPR} records in the blockchain, but only the most recent one is valid.

An R_{ACP} contains a set of access control policies that must be followed by participants in the network. R_{ACP} is defined as a triple (PO, HO, TS), where PO is a set of access control policies; HO is the name of the hospital where the policies need to be enforced; and TS is the time when the policies are created. An R_{ACP} is created when a hospital establishes its access control policies or when the policies are updated. Therefore, each hospital may have multiple R_{ACP} records in the blockchain, but only the most recent one is valid.

An R_{MER} contains the text-based medical reports of a patient and the metadata of the associated multimedia files resulting from a doctor's visit. R_{MER} is defined as 6-tuple (IDS, HO, TXD, MMD, TS, INL), where IDS are the identifications of the patient, nurse, and the doctor who are involved with the doctor's visit; HO is the name of the hospital where the doctor's visit occurs; TXD includes a short summary of the visit and the text-based medical reports; MMD is the metadata of the multimedia files, which includes a description of the multimedia files, the starting location of the actual multimedia file stored, and the sizes of the multimedia files; TS is the time when the record is created; and INL is the index link that points to the nearest previous block that contains an R_{MER} of the same patient. Using the index link, all medical records of a patient are linked together as a singly linked list to facilitate efficient information retrieval of the patient's EHRs. To ensure data security, R_{MER} must be encrypted using the patient's symmetric key SEK stored in the patient's latest R_{UPR}, where SEK is managed by a corresponding β_{SUP}. Note that it is possible the MMD section in an R_{MER} is empty if the record R_{MER} only contains text-based information.

B. Block Structure and Block Generation

A cloud-based block consists of three components, namely the regular component *A*, the multimedia component *B*, and the verification component *C*. Fig. 2 shows an example of a new cloud-based block CB_{h+1}, where *h* is the length of the current blockchain. Component *A* contains the block header and the block records; component *B* holds the actual multimedia files compressed together with their metadata recorded in component *A*; and component *C* contains the hash value of component *A*, i.e., $hash(LB_{h+1})$, the hash value of component *A* and *B*, i.e., $hash(CB_{h+1})$, and a list of digital signatures $ds[CB_{h+1}]_v$, where each peer *v* is an agent β_{SUP} who approves CB_{h+1} during the consensus process. The block header in component *A* contains the hash value of its previous block CB_h, namely $hash(CB_h)$, as well as $hash(LB_h)$ of the last lite block in a LBC, the timestamp when the new block is created, the block ID, and the length *h* of the current blockchain. The block records section contains any number of block records including user profile R_{UPR}, access control policies R_{ACP}, and medical records R_{MER}.

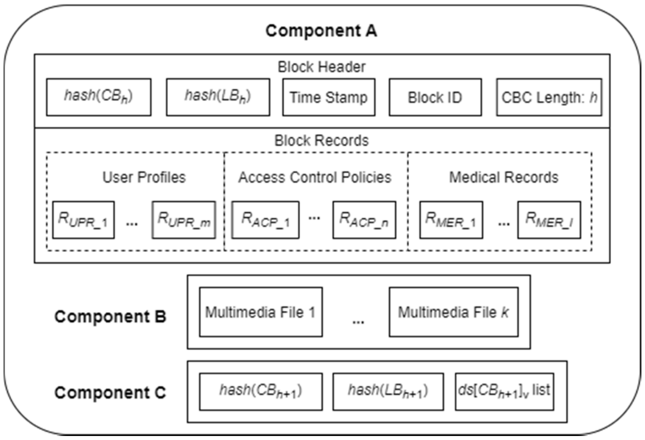

Fig. 2. The structure of a cloud-based block CB_{h+1}

Algorithm 1 shows how a new cloud-based block CB_{h+1} is generated by super peer agent β_{SUP} Ψ. According to the algorithm, agent Ψ first creates an empty cloud-based block CB_{h+1}. It then adds the hash values $hash(CB_h)$ and $hash(LB_h)$, the time stamp, the block ID, and the current blockchain length *h* to the block header of component *A* in CB_{h+1}. After that, it encrypts each multimedia file κ using the patient's symmetric key and adds them to component *B* of CB_{h+1}. The location and the size of each encrypted multimedia file κ are recorded and added to the metadata of the multimedia files in medical record R_{MER}. For each block record φ in the record list Ξ, Ψ processes it according to its record type and adds it to the block records section in component *A* of CB_{h+1}. The symmetric key *SEK* of a regular peer is used to encrypt the block record φ that belongs to the regular peer. Note that items in R_{UPR} must be encrypted individually, so *SEK* can be encrypted differently using the public key of Ψ to make it available to Ψ. Finally, Ψ calculates the hash values $hash(CB_{h+1})$ and $hash(LB_{h+1})$, uses $hash(CB_{h+1})$ to create the digital signature $ds[CB_{h+1}]_\Psi$, and adds all these elements to component *C* of CB_{h+1}.

Once CB_{h+1} has been created, a new lite block LB_{h+1} can be generated from CB_{h+1} by removing its component *B*. In other words, LB_{h+1} is simply a copy of CB_{h+1}'s components *A* and *C* as illustrated in Fig. 2. This allows a regular or super peer agent to store its own copy of LBC without having the need to store the multimedia files. Since component *B* is not included in lite blockchains, validation of a lite block *LB* only requires the calculation of *LB*'s hash value and comparing it with the stored hash value *hash*(*LB*). As each lite block records the hash value of its previous lite block, the lite blockchain is a self-contained blockchain for verification and information retrieval.

Algorithm 1: Generating a New Cloud-Based Block by β_{SUP} Ψ

Input: A list of block records Ξ containing records R_{UPR}, R_{ACP} and R_{MER}, a list of multimedia files Φ, and the most recent cloud-based block CB_h in the blockchain.

Output: A new cloud-based block CB_{h+1} digitally signed by Ψ

```
Create an empty cloud-based block $CB_{h+1}$
Verify and add $hash(CB_h)$, $hash(LB_h)$, time stamp, block ID, and
    current blockchain length $h$ to the block header of $CB_{h+1}$
for each multimedia file $\kappa$ in the list $\Phi$
    Encrypt $\kappa$ and add it to component $B$ of $CB_{h+1}$
    Calculate the location and size info of $\kappa$ in $B$ of $CB_{h+1}$
    Add the above info to the corresponding $R_{MER}.MMD$
for each block record $\varphi$ in the list of records $\Xi$
    if $\varphi$ is an $R_{UPR}$
        if $\varphi.SEK$ is null // a new user
            Generate $\varphi.SEK$ and encrypt it using the public key of $\Psi$
        Encrypt other elements of $\varphi$ and add them to $A$ of $CB_{h+1}$
    else if $\varphi$ is an $R_{ACP}$
        Add it to component $A$ of $CB_{h+1}$
    else if $\varphi$ is an $R_{MER}$
        Encrypt $\varphi$ and add it to component $A$ of $CB_{h+1}$
Calculate $hash(CB_{h+1})$ and $hash(LB_{h+1})$, add them to $C$ of $CB_{h+1}$
Create digital signature $ds[CB_{h+1}]_\Psi$ using $hash(CB_{h+1})$
Add $ds[CB_{h+1}]_\Psi$ to the $ds[CB_{h+1}]_v$ list in $C$ of $CB_{h+1}$
return $CB_{h+1}$
```

V. Data Security and the Consensus Mechanism

A. Role-Based Access Control Policies

There are numerous regular peers participating in the blockchain network, who take different roles such as doctors, nurses and patients. It is critical to specify the appropriate permissions for each role to access the health records stored in blockchains and protect them from being exposed to unauthorized access [15]. In this paper, we define RBAC policies to enforce what data that regular peers, based on their credentials, can access. A regular peer agent β_{REP}, who represents an end user, must seek permission from a super peer agent β_{SUP} for access to sensitive data, as a regular peer agent cannot be trusted to always follow the established policy on its own. Hence, an agent β_{SUP} is responsible for the implementation and enforcement of access control policies for its represented hospital. Note that an agent β_{SUP} enforces the access control policies for both EHRs' text-based information stored in its LBC and multimedia files that can be found in the CBC. A regular peer agent receives permission in the form of a token that is necessary to have in order to access a patient's EHRs stored in the blockchains.

Since there are multiple hospitals in the blockchain network, and each policy record R_{ACP} is a hospital-wide one, a policy record that is applicable to a certain hospital may not be applicable to another one unless it is otherwise specified. Hence, a policy record for a hospital would typically apply to doctors, nurses and patients who are registered with the hospital. Each policy written in a policy record R_{ACP} consists of the following required fields:

- *Hospital*: One or more hospitals that the policy applies to. All involved peers who take the roles in the policy must be registered ones with the specified hospitals.
- *Role*: The roles that one or more regular peers can take.
- *Summary*: A short description of an access control policy.
- *Type*: A policy of operation type specifies a permission of a regular peer to perform a task, while a policy of relationship type specifies the relationship among multiple regular peers.
- *Condition*: The requirements for a permission to be approved or a relationship to be established.
- *Conclusion*: The permission to be granted or the relationship to be established when all conditions are met.

In the following, we show a few examples of RBAC policies that are specified for hospitals H_1 and H_2.

policy PL1 {
  hospital: H_1
  role: patient (Patient *P*)
  summary: Peer with a patient role can access its own medical information
  type: operation
  condition: [Patient *P* is a registered patient in hospital H_1]
  conclusion: Patient *P* is allowed to access Patient *P*'s information in H_1.
}
policy PL2 {
  hospital: H_1
  role: doctor (Doctor *D*), patient (Patient *P*)
  summary: Peer with a doctor role can access its patients' medical information
  type: operation
  condition: [Doctor *D* is a registered doctor in H_1] && [Patient *P* is a registered patient in H_1] && [Doctor *D* is recognized as Patient *P*'s doctor]
  conclusion: Doctor *D* is allowed to access Patient *P*'s information; Doctor *D* is allowed to submit Patient *P*'s new medical information.
}
policy PL3 {
  hospital: H_1
  role: doctor (Doctor *D*), patient (Patient *P*)
  summary: Doctor *D* is the family doctor of patient P in hospital H_1
  type: relationship
  condition: [Doctor *D* is a registered doctor in H_1] && [Patient *P* is a registered patient in H_1]
  conclusion: Doctor *D* is recognized as Patient *P*'s family doctor
}
policy PL4 {
  hospital: H_1, H_2
  role: doctor (Doctor *D*), patient (Patient *P*)
  summary: Doctor *D* (in H_1) is the referral doctor of patient *P* (in H_2)
  type: relationship
  condition: [Doctor *D* is a registered doctor in H_1] && [Patient *P* is a registered patient in H_2] && [Expiration date is 01/02/2023]
  conclusion: Doctor *D* is recognized as patient *P*'s referral doctor
}

Policy PL1 dictates that a peer with a patient role can access a patient's information only if the peer agent is requesting the peer's own information. This policy involves only a single patient as stated by the role component. Policy PL2 dictates that a peer with a doctor role can access a patient's information only if the peer is recognized as a family doctor of the patient and if the peer is the current family doctor of the patient. This policy involves a single patient and a single doctor as stated in the role entry. Policy PL3 establishes the relationship between patient *P* and doctor *D* in hospital H_1. Doctor *D* is recognized as a family doctor of patient *P*, and this relationship is a necessary requirement for other policies that involve doctor *D* and patient *P*, i.e., doctor *D* accessing patient *P*'s information. Policy PL4 establishes another type of relationship between doctor *D* from hospital H_1 and patient *P* from hospital H_2. This policy establishes doctor *D* as a referral doctor of patient *P* with a specified expiration date. The policy is considered invalid once the time has passed the expiration date, and this in effect gives doctor *D* a temporary status as patient *P*'s doctor.

B. Data Encryption and Decryption

Medical records, especially those stored in plain text form in the blockchains run a high risk of having their confidentiality compromised [2]. Thus, all EHRs must be properly secured before they are stored in the blockchains. To this end, we adopt asymmetrical keys and symmetrical keys for user identification as well as data encryption and decryption. Every participating regular peer agent is assigned a pair of public and private keys stored in its R_{UPR}, as described in Section IV.A. These keys are responsible for providing identification via digital signature to other peers in the blockchain network. Every regular peer has a symmetric key *SEK* stored in its R_{UPR}. This key is considered as the secret key used to encrypt and decrypt any data owned or generated from said peer. As mentioned earlier, this key is encrypted by a super peer agent's public key. A regular peer agent must make a request to its super peer agent for the permission to unlock and use this key.

Note that to keep the symmetric key as a secret, it can only be handled by a software agent, who shall never present the key to any end user. Fig. 3 shows the procedure where software agent β_{REP_1} (e.g., a doctor's agent) requests the symmetric key SEK_{REP_2} of β_{REP_2} (e.g., a patient's agent) and any optional multimedia files from super peer agent β_{SUP_A}.

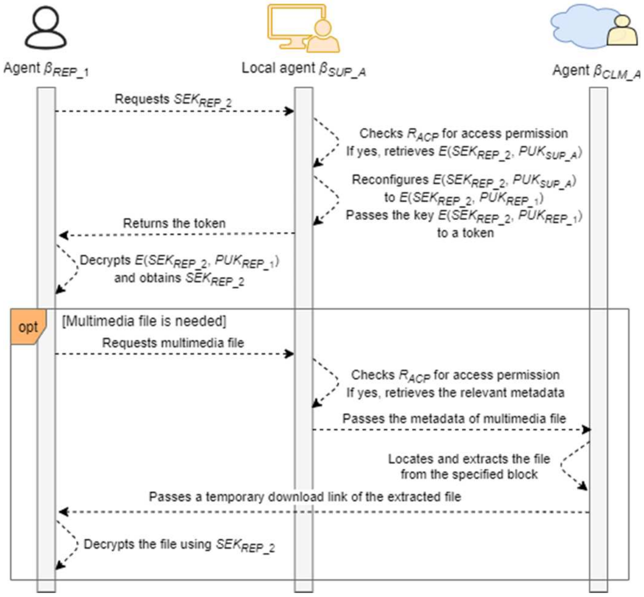
Agent β_{REP_1}
Local agent β_{SUP_A}
Agent β_{CLM_A}
Requests SEK_{REP_2}
Checks R_{ACP} for access permission
If yes, retrieves $E(SEK_{REP\_2}, PUK_{SUP\_A})$
Reconfigures $E(SEK_{REP\_2}, PUK_{SUP\_A})$
to $E(SEK_{REP\_2}, PUK_{REP\_1})$
Passes the key $E(SEK_{REP\_2}, PUK_{REP\_1})$
to a token
Returns the token
Decrypts $E(SEK_{REP\_2}, PUK_{REP\_1})$
and obtains SEK_{REP_2}
opt
[Multimedia file is needed]
Requests multimedia file
Checks R_{ACP} for access permission
If yes, retrieves the relevant metadata
Passes the metadata of multimedia file
Locates and extracts the file
from the specified block
Passes a temporary download link of the extracted file
Decrypts the file using SEK_{REP_2}

Fig. 3. Retrieval of SEK_{REP_2} and multimedia files by agent β_{REP_1}

The process starts when regular peer *REP*_1 needs to access regular peer *REP*_2's medical information. This results in β_{REP_1} making a request to β_{SUP_A} for the permission to use SEK_{REP_2}.

Agent β_{SUP_A} consults the access control policy written in R_{ACP} from β_{SUP_A}'s *LBC*. If approved, β_{SUP_A} retrieves the encrypted key $E(SEK_{REP\_2}, PUK_{SUP\_A})$ from *REP*_2's R_{UPR} stored in the LBC. This key is then decrypted and re-encrypted using β_{REP_1}'s public key PUK_{REP_1} into $E(SEK_{REP\_2}, PUK_{REP\_1})$ and is passed into a token. This token is returned to β_{REP_1}, where the encrypted key is then decrypted into SEK_{REP_2}. In a case when β_{REP_1} also requests to access β_{REP_2}'s multimedia file, β_{SUP_A} retrieves the metadata of the multimedia file from its LBC and sends it to the cloud manager agent β_{CLM_A}. Agent β_{CLM_A} extracts the multimedia file from its CBC and sends a temporary download link of the multimedia file to β_{REP_1} for access. Note that the multimedia file is encrypted using symmetric key SEK_{REP_2}. Since β_{REP_1} has already received SEK_{REP_2} from β_{SUP_A}, β_{REP_1} can immediately decrypt the multimedia file and present it to the end user represented by agent β_{REP_1}.

To showcase how we use these key schemes and the access control policies, we now discuss several typical scenarios.

Scenario 1: Patient *REP*_2 (represented by regular peer agent β_{REP_2}) visits doctor *REP*_1 (represented by regular peer agent β_{REP_1}) for a yearly medical checkup at hospital *A* (represented by super peer agent β_{SUP_A}). This results in several medical data being generated, such as MRI, X-ray, CT scan, ultrasound video, and a doctor's report. Doctor *REP*_1 and patient *REP*_2 both verify all the generated data before sending them to β_{SUP_A}. When agent β_{SUP_A} receives the data, it creates medical record R_{MER}, and places R_{MER} and the multimedia files into a list of block records and a list of multimedia files, respectively, for being included in a new cloud-based block. Note that according to Algorithm 1 (lines 3-15), when the new cloud-based block is generated, both the multimedia files and R_{MER} need to be encrypted using SEK_{REP_2}, retrieved from the encrypted key $E(SEK_{REP\_2}, PUK_{SUP\_A})$ stored in β_{SUP_A}'s LBC.

Scenario 2: Doctor *REP*_1 is a family doctor of patient *REP*_2 at hospital *A*, who needs to retrieve a medical report of *REP*_2 stored in the LBC. Agent β_{REP_1} first sends its credential to super peer agent β_{SUP_A} for the permission to retrieve *REP*_2's symmetric key SEK_{REP_2}. A similar process shown in Fig. 3 up until the *opt* section then occurs. A token containing SEK_{REP_2} is sent to agent β_{REP_1} allowing it to decrypt and access the medical report of *REP*_2 stored in the LBC.

Scenario 3: Doctor *REP*_1 from hospital *A* needs to access a multimedia file that are generated during patient *REP*_2's visit at hospital *A*. Similar to Scenario 2, β_{REP_1} first sends its credential to super peer agent β_{SUP_A} for the permission to view *REP*_2's medical information. If *REP*_1 has the needed permission, it receives symmetric key SEK_{REP_2} allowing it to retrieve *REP*_2's medical records in LBC. However, since the multimedia file is stored in CBC rather than LBC, β_{SUP_A} retrieves the metadata of the multimedia file from its LBC, and sends it to the cloud manager agent β_{CLM_A}. As shown in Fig. 3, β_{CLM_A} extracts the file from its CBC based on the given metadata. A temporary link is then created and passed back to β_{REP_1}. Agent β_{REP_1} can then use the link to download the multimedia file from the cloud, and decrypt it using SEK_{REP_2}.

Scenario 4: Doctor *REP*_1 from hospital *A* refers patient *REP*_2 to doctor *REP*_3 from hospital *B*. Agent β_{REP_1} makes a request to its local agent β_{SUP_A} to allow doctor *REP*_3 from hospital *B* to access patient *REP*_2's medical records. This involves β_{REP_1} sending a new access control policy to β_{SUP_A}. The super peer agent β_{SUP_A} verifies the new policy and submits a new block record R_{ACP} for approval by the consensus process. Once the new access control policy is approved and added to the blockchains, agent β_{REP_3} has the necessary credential to make a request to β_{SUP_A} to access *REP*_2's information. Note that β_{REP_3} must make a request to β_{SUP_A} instead of its own super peer agent β_{SUP_B} to retrieve the symmetric key SEK_{REP_2} as the symmetric key is encrypted using PUK_{SUP_A}.

C. The Consensus Process

The consensus process in our approach requires a majority approval from the super peer agents participating in the blockchain network. Let λ be the total number of super peer agents participating in the network, and the number of super peer agents required to approve a new block is $\lambda/2$, excluding the block announcer – the super peer agent who initiates the consensus process by broadcasting a new block to the rest of the super peer agents. We call the block announcer the primary super peer agent or β_{SUP_P} (denoted as $\Psi0$). Fig. 4 shows an example of the consensus process with three super peer agents, β_{SUP_P} ($\Psi0$), β_{SUP_1} ($\Psi1$), and β_{SUP_2} ($\Psi2$). In this example, each super peer agent represents a hospital, which has a private cloud managed by a cloud manager agent, and a regular peer agent registered with the hospital. The three hospitals are *Hospital*_0 ($\Psi0$, β_{CLM_0}, β_{REP_0}), *Hospital*_1 ($\Psi1$, β_{CLM_1}, β_{REP_1}), and *Hospital*_2 ($\Psi2$, β_{CLM_2}, β_{REP_2}).

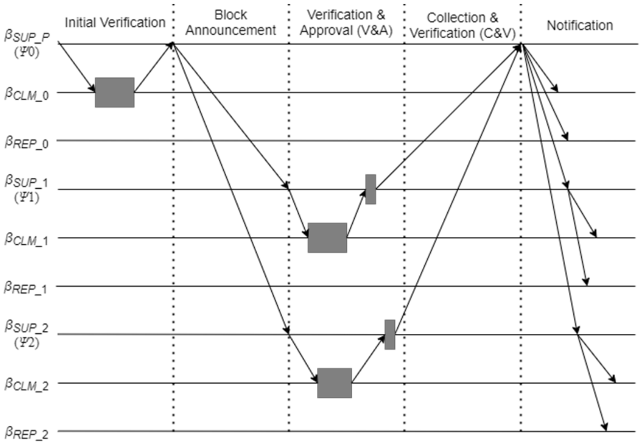

Fig. 4. An illustration of the consensus process

According to Fig. 4, the consensus process is divided into five phases, namely *Initial Verification*, *Block Announcement*, *Verification & Approval* (*V&A*), *Collection & Verification* (*C&V*), and *Notification*. Initially, a primary super peer agent β_{SUP_P} ($\Psi0$) creates a new cloud-based block CB_{h+1} and signs the block with its digital signature $ds[CB_{h+1}]_{\Psi0}$. This block is sent first to the cloud manager agent β_{CLM_0} for verification purposes in the *Initial Verification* phase. During this stage, β_{CLM_0} checks CB_{h+1}'s integrity by verifying the signature $ds[CB_{h+1}]_{\Psi0}$. If any fault is found, β_{CLM_0} informs $\Psi0$ of the error and aborts the process. Otherwise, β_{CLM_0} informs $\Psi0$ to proceed. Agent $\Psi0$ then generates a lite block LB_{h+1} from CB_{h+1} and checks its validity. Agent $\Psi0$ announces CB_{h+1} and LB_{h+1} to β_{SUP_1} ($\Psi1$) and β_{SUP_2} ($\Psi2$) for downloading during the *Block Announcement* phase. In the *V&A* phase, agents $\Psi1$ and $\Psi2$ send CB_{h+1} to their own private cloud manager agents, β_{CLM_1} and β_{CLM_2}, where a similar process in the *Initial Verification* phase occurs. If no error is detected, $\Psi 1$ and $\Psi 2$ create their digital signatures, $ds[CB_{h+1}]_{\Psi 1}$ and $ds[CB_{h+1}]_{\Psi 2}$, respectively, and send them back to $\Psi 0$. In the *C&V* phase, $\Psi 0$ waits for digital signatures sent by $\Psi 1$ and $\Psi 2$ and verifies them when received. If a digital signature is valid, $\Psi 0$ adds it to the list of digital signatures $ds[CB_{h+1}]_{\text{list}}$. When $\Psi 0$ has collected at least $\lambda/2$ valid digital signatures, it sends them to $\Psi 1$ and $\Psi 2$ in the *Notification* phase. Upon receiving the list of digital signatures, all super peer agents add it to their copies of CB_{h+1} and LB_{h+1}. The completed CB_{h+1} and LB_{h+1} can now be added to their CBCs and LBCs, respectively. Afterwards, all super peer agents broadcast the completed LB_{h+1} to their respective regular peer agents β_{REP_0}, β_{REP_1}, and β_{REP_3} for LBC updating.

VI. Case Study

To demonstrate the feasibility and efficiency of our proposed approach, we conduct experiments to simulate cloud and lite blockchains and evaluate their performance. The experiment environment consists of multiple identical computers connected under the same domain network. The computer specifications are Intel® Core™ i7-6700k CPU @ 3.40GHz (4 CPU Cores); 16 GB RAM, Windows 10 OS (64-bit, x64-based processor); and 512GB SSD Hard Drive. The domain network used in this case study has a recorded speed of 680 Mbps.

A. Block Sizes of CBCs

Large sizes of cloud-based blocks present major challenges for maintaining CBCs in the clouds. In this experiment, we generate dynamic numbers of blocks per day based on the number of patient visits to a simulated hospital during that day. The frequencies of such visits are randomly generated with a range of [1, 1000]. A patient's visit results in generating EHR that contains text-based information and optionally multimedia files such as images and videos. We assume for each patient's visit, in addition to text-based reports, there is less than 10% probability of producing 1 to 10 image files (e.g., X-ray images) and less than 10% probability of producing 1 to 4 video files (e.g., ultrasound videos). We also assume that the size of a high-quality image file is about 1-4 MB, a high-quality video file is about 10-50 MB, and a text-based report is about 3-7 KB. Fig. 5 shows the sizes of cloud-based blocks that can be generated during a day under the above settings. In the figure, we compare our dynamic approach with two simple approaches, namely "One Block" approach that stores all data generated by patients during a day in a single block, and "Two Blocks" approach that divides generated patients' data into two blocks. Only one single consensus process per day is required for each hospital under the "One Block" approach; however, the size of the resulting block may go over 6GB when the number of patient visits reaches 1000. Similarly, under the "Two Blocks" approach, two consensus processes per day are required for each hospital; however, the size of the resulting block may still go over 3GB when the number of patient visits reaches 1000. To further reduce the sizes of cloud-based blocks, in the third approach, we generate dynamic numbers of blocks per day based on the predicted number of daily patient visits. When the number of patient visits is less than 200, we create one block, and we create two blocks when the number of patient visits is between 201 to 400, and so on. When the number of patient visits reaches 1000, we generate up to 5 blocks per day. From the figure, we can see that we are able to keep a consistent block size of around 1 GB with the cost of increased number of blocks per day when the number of patient visits is high. However, this is well worth the tradeoff as the cloud-based blocks become much easier to be maintained in the clouds and accessed by the regular peers within the blockchain network.

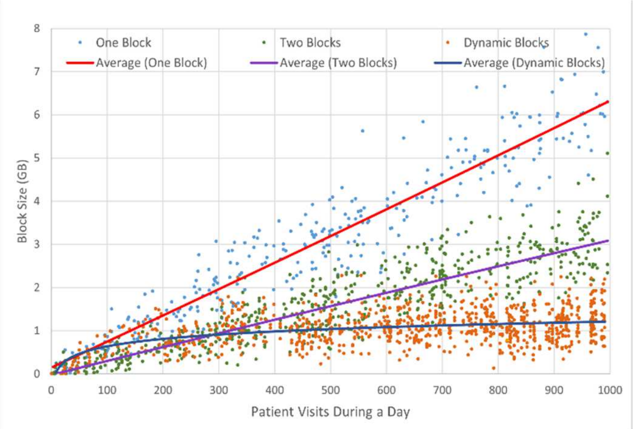

Fig. 5. Cloud-based blocks generated during a day

B. Search Time in an LBC

As defined in Section IV.A, index link $R_{MER}.INL$ allows all R_{MER} that belong to the same patient to be linked together. In this experiment, we show the index links greatly shorten the search time for a particular patient's EHRs in an LBC when multiple years EHRs are searched. This experiments are designed to search for patient's text-based information in a simulated LBC containing 10 years' worth of patient's information. We conduct multiple experiments, with each having 10 super peer agents participating in the blockchain network. Each super peer agent represents a small, medium, or large hospital in the network, which has a range of [1, 100], [1, 500], and [1, 1000] daily patient visits, respectively. We record the average search time vs. the maximum number of years searched in each experiment. To compare our index-link-based approach with multi-threaded sequential searches, we show the experimental results together as in Fig. 6. The figure shows the average search time for all EHRs of a random patient in respect to the number of most recent years to be searched. From the figure, we can see our index-link-based approach outperforms the sequential search approaches for average search time. The sequential search approaches use 4, 8, and 16 multiple threads to allow faster search time; however, the average search time of those approaches increases dramatically when the number of search years increases due to the limited number of available CPU cores on each computer and the usage of the same hard disk to store all blocks of a LBC. In contrast, the index-link-based approach uses only a few seconds to search for all relevant EHRs that belong to a patient. To achieve this high performance, we also conduct a one-time full pre-scan to build a separate index file that records the ID of the block that contains a patient's latest R_{MER}. Note that the full pre-scan only needs to be performed once, and after that, an update can be done when a new block is added to the blockchain.

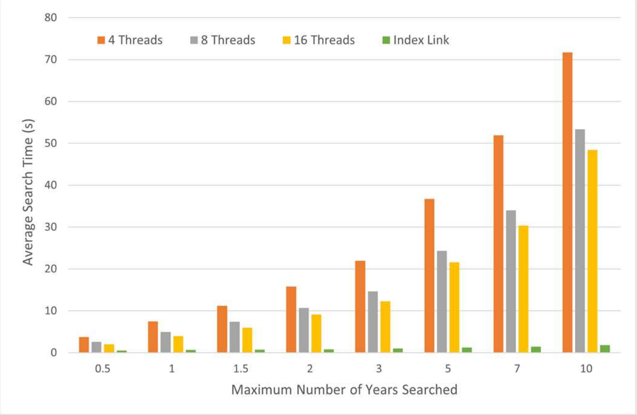

Fig. 6. Average search time for patient's information in an LBC

C. *Retrieval Time of Multimedia Files in a CBC*

In this experiment, we record and analyze the retrieval time of multimedia files from a CBC. The retrieval time includes the time needed to extract multimedia files from a CBC, download the files from the cloud, and decrypt the files by a regular peer agent. We adopt the same experiment settings as in Section VI.B. Table I shows the average number of blocks where multimedia files are found for a typical patient during a certain number of years. The table also shows the upper range of the number of blocks containing multimedia files with 95% of the patients within the range, as well as the average total size of all multimedia files for one patient during a number of years.

TABLE I. NUMBER OF BLOCKS AND AVERAGE SIZES VS. NUMBER OF YEARS

Max years	Average number of blocks containing multimedia files	Upper range of the number of blocks with multimedia files for 95% patients	Average total size of multimedia files (GB)
1	1.17	3	0.192
2	2.38	5	0.302
3	3.71	7	0.437
4	4.76	8	0.537
5	6.02	10	0.685
6	7.20	11	0.767
7	8.50	12	0.815
8	9.56	14	0.899
9	10.84	15	0.973
10	12.12	17	1.088

Fig. 7 shows the average time needed to retrieve a patient's multimedia files from the cloud based on the upper range of the number of blocks containing multimedia files with 95% of the patients within the range, as shown in Table I. In the experiment, a super peer agent β_{SUP} first collects the metadata of the multimedia files from a varying number of patient *REP*_2's R_{MERs} stored in the respective blocks. It then sends the set of metadata along with the credential of the requesting agent β_{REP_1} (e.g., an agent representing a doctor) to the cloud manager agent β_{CLM}. If permission is granted by agent β_{CLM}, β_{CLM} extracts all requested multimedia files from its CBC and sends the temporary download links of the files to β_{REP_1} for file downloading. Finally, the downloaded multimedia files are decrypted by β_{REP_1} and presented to the end user *REP*_1. The above processes are timed under the "Extract", "Download", and "Decode" sections in Fig. 7. We can see that with a typical worst case with 20 blocks containing multimedia files, the average retrieval time is 22 seconds, which is acceptable. In addition, we notice that the most significant portion of the retrieval time is to download the multimedia files, which can be improved by having a better network bandwidth.

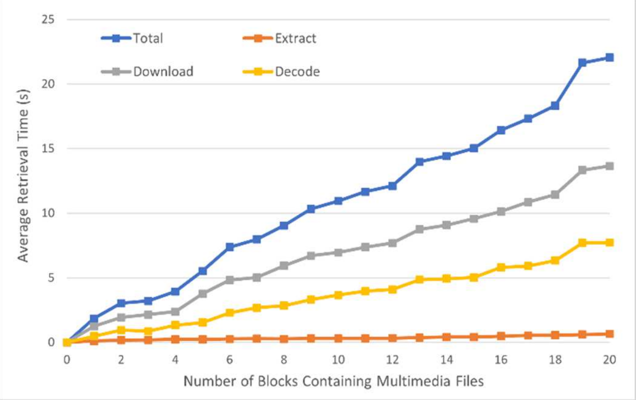

Fig. 7. Average time to retrieve a patient's multimedia files from the cloud

D. *Consensus Latency*

In our last experiment, we measure the performance of the consensus process based on the number of super peer agents participating in the blockchain network. The timing of this consensus process begins when the primary super peer agent $\Psi 0$ verifies and digitally signs a new block in the *Initial Verification* phase, and ends after $\Psi 0$ sends the list of digital signatures to all other super peer agents as well as $\Psi 0$'s regular peer agents in the *Notification* phase. To simplify matters, the timing does not include the amount of time used by other super peer agents to notify their respective regular peer agents to update their blockchains. As shown in Fig. 8, we test the consensus process using fixed block sizes of 0.5GB, 1GB, 1.5GB, and 2GB. We can see that the average consensus time increases faster for a new block with a larger size. This is due to more disseminating time required to share a larger block with other participants in the blockchain network. Furthermore, in all cases, the average consensus time increases significantly when the number of participating super peer agents also increases. While not shown in Fig. 8, the timing of the consensus process includes the time for broadcast, file downloading, and block verification. The broadcast part involves the time for the exchange of notifications and digital signatures between $\Psi 0$ and other super peer agents. The file downloading part involves the time needed for all super peer agents to retrieve the new cloud-based block from $\Psi 0$'s private cloud to their own respective private clouds. The block verification part involves the average time taken for a super peer agent to verify the validity of a new block. The main reason as to why the consensus time seems to increase dramatically with the increased number of super peer agents is mainly due to the time needed to download the new block. The consensus process requires various super peer agents to download the new cloud-based block from $\Psi 0$'s private cloud. Increasing the number of super peer agents inevitably leads to an increase of network congestion, and consequently, an overall longer time is required for the acquisition of the new block for all participating super peer agents. On the other hand, the time required for broadcast and block verification is not significantly impacted by the total number of super peer agents as those tasks are performed concurrently. Based on the experimental results, we can conclude that although our approach might not work well with a large number of participating super peer agents, it demonstrates reasonable average consensus time for a limited number of super peer agents located within a local area, such as a city and urban areas with no more than 30 participating hospitals.

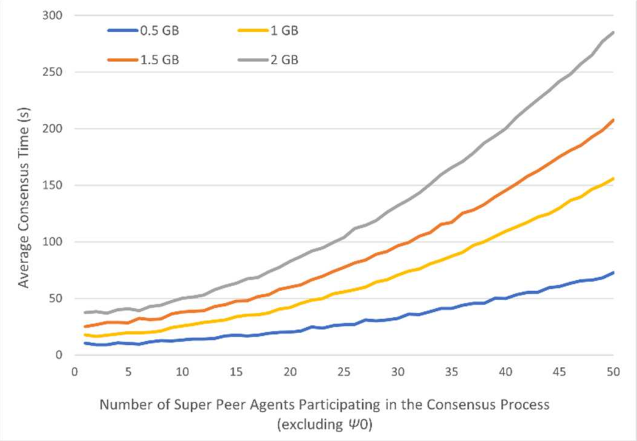

Fig. 8. Consensus time needed for varying numbers of super peer agents

VII. Conclusions and Future work

Blockchain technology is a promising field of research with many potential applications for real world scenarios, particularly for data storage and sharing of sensitive information among multiple entities. In this paper, we explore on a novel approach to storing and sharing EHRs in a cross-hospital setting using the blockchain technology. We introduce a cloud-based blockchain (CBC) along with a lightweight version called lite blockchain (LBC). Both CBCs and LBCs are synchronized to be identical, except for big data that are stored exclusively in CBCs. The use of CBCs to handle storage and sharing of big data allows regular peers, e.g., doctors and patients, to participate in the blockchain network via their LBCs without the need for significant investment in network bandwidth, and memory and storage space. The experimental results show that our cloud-based blockchain approach has reasonable performance for secure and reliable big data storage service designed for a healthcare system within a local area.

In our current approach, we assume all super peer agents can be trusted in the blockchain network as they represent healthcare organizations; meanwhile, the consensus process heavily relies on their responsible collaborations on verifying new blocks as well as ensuring the validity of the digital signatures provided by super peer agents. In future work, we plan to design a more advanced consensus mechanism that also takes malicious organizations into consideration. Furthermore, we plan to improve our current approach using a hierarchical architecture for better scalability with more healthcare organizations involved, and conduct performance comparison with existing off-chain methods such as IPFS [10]. This would allow further applications of our approach to a wider range of areas such as statewide and countrywide. Finally, we will explore different domains including smart contracts for real estates and electronic student records (ESR) management, with more efficient and effective encryption procedures in our cloud-based blockchain approach. Such an improved approach will lead to a more secure and reliable way to store and share sensitive big data in various cloud-based applications.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2009. Retrieved on March 5, 2021 from https://bitcoin.org/bitcoin.pdf

[2] T. Kumar, V. Ramani, I. Ahmad, et al., "Blockchain Utilization in Healthcare: Key Requirements and Challenges," In *Proceedings of the 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Ostrava, Czech Republic, Sept. 17-20, 2018, pp. 1-7.

[3] IBM, "Why New Off-Chain Storage is Required for Blockchains," Document Version 4.1, *IBM Storage Reprot*, 2018. Retrieved on March 8, 2021 from https://www.ibm.com/downloads/cas/RXOVXAPM

[4] R. Kumar, N. Marchang and 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)*, Bengaluru, India, Jan. 7-11, 2020, pp. 1-5.

[5] E. AbuKhousa, N. Mohamed and J. Al-Jaroodi, ''E-Health Cloud: Opportunities and Challenges,'' *Future Internet*, Special Issue: Future e-Health, Vol. 4, No. 3, 2012, pp. 621-645.

[6] M. T. de Oliveira, L. H. A. Reis, R. C. Carrano et al., "Towards a Blockchain-Based Secure Electronic Medical Record for Healthcare Applications," In *Proceedings of the 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, May 2019, pp. 1-6.

[7] S. Alexaki, G. Alexandris, V. Katos et al., "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)*, Barcelona, Sept. 2018, pp. 1-6.

[8] H. Zhao, P. Bai, Y. Peng and R. Xu, "Efficient Key Management Scheme for Health Blockchain," *CAAI Transactions on Intelligence Technology*, Vol. 3, No. 2, June 2018, pp. 114-118.

[9] R. Ming and H. Xu, "Timely Publication of Transaction Records in a Private Blockchain," In *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, IEEE BSC 2020, Macau, China, December 11-14, 2020, pp. 116-123.

[10] R. Kumar and R. Tripathi, "Implementation of Distributed File Storage and Access Framework Using IPFS and Blockchain," In *Proceedings of the Fifth International Conference on Image Information Processing (ICIIP)*, Shimla, India, Nov. 2019, pp. 246-251.

[11] H. Wang and Y. Song, "Secure Cloud-Based EHR System Using Attribute-Based Cryptosystem and Blockchain," *Journal of Medical Systems*, Vol. 42, Article number: 152, August 2018, pp. 1-9.

[12] H. Guo, W. Li, M. Nejad and C. Shen, "Access Control for Electronic Health Records with Hybrid Blockchain-Edge Architecture," In *Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain)*, Atlanta, GA, USA, 2019, pp. 44-51.

[13] H. Guo, W. Li, E. Meamari, C. Shen and 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)*, Toronto, ON, Canada, may 2020, pp. 1-5.

[14] D. C. Nguyen, P. N. Pathirana, M. Ding and A. Seneviratne, "Blockchain for Secure EHRs Sharing of Mobile Cloud Based E-Health Systems," *IEEE Access*, Vol. 7, 2019, pp. 66792-66806.

[15] M. Meingast, T. Roosta and 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*, New York, NY, USA, Aug. 2006, pp. 5453-5458.