d-MABE: Distributed Multilevel Attribute-Based EMR Management and Applications

Ehab Zaghloul, Tongtong Li, *Senior Member, IEEE*, Matt W. Mutka, *Fellow, IEEE*, and Jian Ren, *Senior Member, IEEE*

Abstract—Current systems used by medical institutions for the management and transfer of Electronic Medical Records (EMRs) can be vulnerable to security and privacy threats. In addition, these systems are centralized, often lack interoperability, and give patients limited or no access to their own EMRs. In this article, we propose a novel distributed data sharing scheme that applies the security benefits of blockchain to address these concerns. We deploy smart contracts on Ethereum blockchain and utilize a distributed storage system to alleviate the dependence on the record-generating institutions to manage and share patient records. To preserve privacy of patient records, we implement our smart contracts as a method to allow patients to verify attributes prior to granting access rights. Our proposed scheme also facilitates selective sharing of medical records among staff members that belong to different levels of a hierarchical institution. We provide extensive security, privacy, and evaluation analyses to show that our proposed scheme is both efficient and practical.

Index Terms—EMRs, distributed data sharing, Ethereum blockchain, smart contract, symmetric, and attribute-based encryption

1 Introduction

Record-sharing between medical institutions can help improve medical diagnoses, treatment decisions, and enhance the overall patient experience. However, the current methods used to manage and share medical records can be inefficient or limited due to the lack of interoperable systems. Institutions may integrate computerized systems to store EMRs, however, in most cases, these systems are not designed to communicate with one another resulting in additional overhead costs when sharing patient records. In addition, patients generally must rely on the record-generating institution to do the sharing in an efficient, secure, and private way. This incurs additional computational costs to safely store, maintain, and transmit records when requested by other institutions. Furthermore, some nations may have insufficient legal requirements associated with sharing of patient records which makes patient data especially vulnerable to security and privacy threats. In the case of a nation with a central healthcare network, the systems employed may be mismanaged, antiquated, and potentially jeopardize patient data. Therefore, there has been an effort to develop comprehensive, secure, and private electronic record sharing systems that also eliminate reliance on the record generating institution.

In the U.S., the Health Insurance Portability and Accountability Act of 1996 (HIPAA) [1] sets a guideline for secure and private use of patient health data. It does not define, however, how these systems are to be developed and applied. As a result, many centralized and private applications have come to market in the U.S. as ways to manage EMRs. Because these systems are often built on proprietary technology, lack of interoperability between medical institutions is a major concern. Meaning that, in majority, the burden of transferring health records lies on patients, often times in printed copies from one medical institution to another. In other cases, patient records are transferred via fax or systems similar to electronic mail. These common transfer methods can be inefficient and possibly jeopardize patient privacy and data security. There is also a lack of an auditable trail of data transfer, and there is no way to preclude who will view a patient record on the other end of a fax. Lastly, in cases where there is no likely method of data transfer or data cannot be trusted, physicians may resort to duplicate testing causing resource waste.

Other nations with universal healthcare may use a national system, such as the National Health Service (NHS) [2] in the United Kingdom. The NHS serves as a centralized national authority allowing patients and medical institutions to share records efficiently. Many features of the NHS system provide advantages that address the challenges of record-sharing seen in the U.S. health system. As part of the NHS, patients have free access to a service named the Summary Care Record (SCR) [3]. This service can provide medical institutions essential information of the patient in cases where the patient is being treated by someone other than their General Practitioner (GP). The NHS provides patients some control over how their data is shared with the ability to opt-out of the SCR using an SCR opt-out form. However, while the NHS system may address some challenges with sharing patient data effectively, the use of a centralized system presents security and privacy complications. To complete the SCR opt-out form, a patient must fill out the form and send it to their GP's practice in either paper or digital format. Depending on the administrative practices of the GP, action in response to the form may take days to weeks

• *The authors are with Michigan State University, East Lansing, MI 48824-1226 USA. E-mail: {ebz, tongli, mutka, renjian}@msu.edu.*

Manuscript received 10 July 2019; revised 29 May 2020; accepted 12 June 2020. Date of publication 18 June 2020; date of current version 15 June 2022. (Corresponding author: Jian Ren.)
Digital Object Identifier no. 10.1109/TSC.2020.3003321

to complete. In addition, since the NHS is a central entity when compared to individual medical institutions in the US system, a malicious actor could potentially access more patient EMRs if the system becomes compromised.

In this paper, we propose a novel distributed record sharing scheme that leverages blockchain and smart contracts to eliminate the record-generating institution from the record-sharing process. Blockchain technology was first introduced in 2009 with the evolution of the digital cryptocurrency, Bitcoin [4], as a result of growing security concerns associated with the reliance on a central party. One of the main goals of this project was to eliminate the need of a trusted third party, such as banks, to facilitate payments between individuals. Shortly after the advent of Bitcoin, based on the concept initially introduced by Nick Szabo in 1994 [5], smart contracts were incorporated into blockchain-based systems. Smart contracts are self-executing pieces of code that can digitally enforce and facilitate verified negotiations of contracts between two participating individuals without the need of a trusted third party. They continue to appear in today's systems such as Ethereum [6], which aim to provide services beyond simple payments. Using such smart contracts, users can design and customize their own systems on top of blockchains such as Ethereum. Similar to payment transactions, the deployment and execution of smart contracts is immutable, irreversible and can be tracked over the public blockchain. Therefore, smart contracts are well suited to facilitate data management and sharing. They can be used by data owners to define access control mechanisms that grant certain data users access to the data.

Our proposed work aims at solving the security and privacy challenges and interoperability issues of the current solutions. In the existing systems, EMRs are stored within the generating institution, which significantly limits the patients' access to their own records. They must also trust that the institutions will not leak any sensitive information of their EMRs to unintended organizations. In comparison, our proposed work empowers them over their EMRs. First, it provides record ownership to the patients, allowing them to be in actual possession of their records, rather than have them stored by the generating institution. Second, it allows patients to selectively share different parts of their records with distinct data users based on their privacy preferences.

The work presented in this paper expands significantly on the model presented in [7]. In this model, patients have no method to monitor access to their EMRs. They must rely on trusted third parties to efficiently share this information. In comparison, our proposed scheme is distributed in terms of data management and storage, and grants patients control over their records, thereby minimizing the dependency on the trusted third parties. Patients can also independently trace the history of access to their EMRs. The main contributions of this paper can be summarized as follows:

1) We develop a novel distributed electronic medical record-sharing scheme for patients that is secure, preserves the privacy of patient data, and is more efficient than traditional paper record sharing.
2) We propose a distributed method for verifying medical institution staff member attributes over the blockchain before issuing them access keys. This method can help minimize blind reliance on key-issuers whose behavior cannot be verified when validating attributes of staff members.
3) We conduct comprehensive security and privacy analyses of the proposed scheme.
4) We implement our proposed scheme using smart contracts and deploy them over the Ethereum blockchain for performance evaluation and empirical results.

The rest of this paper is organized as follows. In Section 2, the related work is reviewed. In Section 3, preliminaries are introduced that summarize key concepts used in this research. Next, in Section 4, the problem formulation is described, outlining the system model and our design goals. In Section 5, our proposed scheme is presented in detail, outlining the proposed algorithms. Following that, in Section 6 we formally prove the security of our proposed scheme. In Section 7, we present a performance and application analysis that is supported with numerical results. Finally, in Section 8, a conclusion is drawn to summarize our research.

2 Related Work

In 2015, a decentralized data management scheme was introduced that facilitated access-control management over a blockchain [8]. In this system, the actual data records are stored in an off-blockchain storage while pointers to these records are maintained by a key-value storage over the blockchain. This solution helps simplify the amount of data processed on the blockchain. However, the method used to define access policies in this scheme does not consider hierarchical data sharing. A user that desires to share files selectively among multiple users in a hierarchy will have to define several access policies that could become a complex problem as the number of users increases drastically.

In 2016, MedRec [9], the first functional electronic medical record-sharing system built on some concepts from [8] was introduced. This work builds on three Ethereum [6] smart contracts that manage the authentication, confidentiality, and accountability during the data sharing process. In this system, the primary entities involved in maintaining the blockchain are the parties interested in gaining data, such as researchers and public health authorities. In return, the institutions are rewarded with access to aggregate and anonymized data. However, the success of such a system is dependent on the participation of entities that maintain the system in return for data. Similar to [8], MedRec does not consider hierarchical data sharing.

In 2017, another functioning electronic medical record-sharing scheme was presented to provide a secure solution using blockchain [10]. However, the system uses a cloud-based storage system to store medical records. With centralized storage, the system becomes liable to a single point of failure. Moreover, similar to MedRec, this work builds over a private and monitored blockchain where each node involved in maintaining consensus is known. In comparison, our proposed work eliminates the need for centralized storage by relying on distributed storage such as IPFS. It also builds over a permissionless blockchain instead of a private one, mitigating the reliance on predefined validating nodes in a private blockchain setting. Moreover, in contrast to both schemes, we also present a security analysis and show that our proposed is secure against potential attacks.

In 2018, a study was conducted to discuss several open problems in order to use blockchains for data management and analytics [11]. The study shows that more research is required in order to leverage the capabilities of current data management systems into blockchain-based models. In addition, the study reflects the importance of improving the security and privacy countermeasures.

3 Preliminaries

3.1 Ciphertext Policy Attribute-Based Encryption

Ciphertext Policy Attribute-Based Encryption (CP-ABE) is a fine-grained access control and encryption scheme. It allows users to share their data selectively by using access policies that are integrated into the ciphertexts during encryption. CP-ABE formally divides the process into four main functions.

(i) $\mathsf{Setup}(1^\kappa)$: a probabilistic function with input security parameter κ. The function is performed by a key-issuer to generate a public key PK and the master key MK.
(ii) $\mathsf{KeyGeneration}(\mathsf{MK}, \mathbb{A})$: a probabilistic function carried out by a key-issuer to generate a unique secret key SK for a data user based on the MK and a set of attributes $\mathbb{A} = \{\mathsf{A}_1, \mathsf{A}_2 \ldots, \mathsf{A}_n\}$ possessed by the user.
(iii) $\mathsf{CPABE\text{-}Enc}(\mathsf{PK}, \mathsf{m}, \mathcal{T})$: a probabilistic function carried out by the data owner to encrypt message m under an access policy $\mathcal{T}$ and generate ciphertext CT.
(iv) $\mathsf{CPABE\text{-}Dec}(\mathsf{CT}, \mathsf{SK})$: a deterministic function carried out by the data user to decrypt a ciphertext CT using the uniquely generated secret key SK for the user.

3.2 Privilege-Based Multilevel Organizational Data-sharing

Privilege-based Multilevel Organizational Data-sharing scheme (P-MOD) [12] is an extension of the CP-ABE that handles data sharing in complex hierarchical organizations. P-MOD partitions a data file into multiple segments based on user privileges and data sensitivity. Each segment of the data record is shared according to user privileges and the set of attributes that satisfy the hierarchical access policies.

Privilege-Based Access Structure. The privilege-based access structure divides the data users of an organization into a hierarchy that consists of k levels, $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_k\}$. Each level is associated with an access policy, $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$. An access policy $\mathcal{T}_i$ specifies a set of rules defined using logic gates and the different sets of attributes that can satisfy these rules. Data users in possession of a correct set of attributes that can satisfy a certain access policy $\mathcal{T}_i$ belong to level $\mathcal{L}_i$.

Data Partitioning and Encryption. The data owner partitions a data file $\mathcal{F}$ into a set of k record segments, that is $\mathcal{F} = \{\mathsf{F}_1, \mathsf{F}_2, \ldots, \mathsf{F}_k\}$. Segment F_1 contains the most sensitive information of $\mathcal{F}$ that is to be shared with data users belonging to level $\mathcal{L}_1$. Segment F_k contains the least sensitive information of $\mathcal{F}$ that is to be shared with all data users belonging to any level. Next, the data owner generates a set of secret keys $\{\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_k\}$ to encrypt the corresponding segments of $\mathcal{F}$. The key sk_1 is randomly selected by the data owner. The remaining keys are then derived using a cryptographic hash function Hash as follows

$$\mathsf{sk}_{i+1} = \mathsf{Hash}(\mathsf{sk}_i). \tag{1}$$

A privileged data user that satisfies access policy $\mathcal{T}_i$ belongs to level $\mathcal{L}_i$ and is granted key sk_i. Given sk_i, the data user can derive keys $\{\mathsf{sk}_{i+1}, \ldots, \mathsf{sk}_k\}$ belonging to levels $\{\mathcal{L}_{i+1}, \ldots, \mathcal{L}_k\}$ as defined in Equation (1). However, given the properties of Hash function, sk_i cannot be used to derive any of the keys $\{\mathsf{sk}_1, \ldots, \mathsf{sk}_{i-1}\}$. Each $\mathsf{F}_i \in \mathcal{F}$ is then encrypted with its corresponding generated secret key sk_i. Finally, each sk_i is encrypted with CP-ABE under its corresponding access policy $\mathcal{T}_i$.

3.3 Blockchain

A blockchain is a public ledger that stores all the cryptographically processed transactions performed over a peer-to-peer (P2P) network. For presentation purposes, we refer to Ethereum, an open source blockchain-based network that provides its users with a platform to build, deploy and run decentralized applications [13]. The P2P network nodes offer a decentralized Turing-complete virtual machine named the Ethereum Virtual Machine (EVM). It can execute smart contracts deployed by the users over the public and non-trusted network nodes. In addition, log entries may be contained in receipts which are attached to transactions executed by the EVM. These logs represent the results of *events* fired from the smart contracts.

3.4 InterPlanetary File System

InterPlanetary File System (IPFS) [14] is a peer-to-peer network used to store and share content-addressable data over the network nodes in a distributed manner. The network runs a stack of protocols that are responsible for storing and accessing the data stored over it.

Identities. To join the P2P network, nodes first generate unique node identities before establishing connections.

Network. The network layer in IPFS manages the established connections between the connected peers.

Routing. The routing layer keeps track of the addresses of nodes within the network and the data they store. It uses a Distributed Hash Table (DHT) to identify the data stored over any node of the network.

Exchange. IPFS uses BitSwap, a data exchange protocol inspired by BitTorrent [15], to exchange data between nodes.

Objects. IPFS partitions data files into segments and cryptographically hashes each segment, where the hashes are used to reference the location of the corresponding segments.

4 Problem Formulation

Upon visiting a medical institution, a patient interacts with a wide spectrum of distinct staff members that may include but is not limited to: admittance staff members, physicians assisted by nurses and technicians that provide the required treatment, and maybe even financial or discharging staff members. If those staff members can individually and efficiently access the necessary parts of previous medical record(s) of the patient generated by other institutions required to perform their specific jobs, they can provide the patient with expedited admittance and enhanced diagnosis and treatment decisions, reducing the overall time spent during the visit. For example, an ER physician may learn severe drug allergies or current medication for an

unresponsive patient being rushed into the ER from his/her previous records. As a result, the physician can prevent iatrogenic illnesses caused by administering inappropriate medication to that patient or harmful drug interactions. By accessing critical medical information, patient safety is enhanced while saving time and resources. However, for the sake of patient privacy, only necessary staff members should be given permission to access patient record(s) from the medical history determined by the profile of the patient.

In this paper, we denote a record generated by an institution for a patient as $\mathcal{R}$. We denote the record attributes as $\{\mathsf{R}_j \in \mathcal{R} \mid 1 \leq j \leq n\}$, where n is the maximum number of attributes within the record. Record attributes may include previous medical history, medical treatment, lab tests, medication, personal information, financial statements, credit card information, and others. The corresponding attribute values of the record generated for the patient can be denoted as $\{a(\mathsf{R}_j) \mid 1 \leq j \leq n\}$.

In current systems, records often have a variety of attributes and may be blocked or intentionally delayed by competitive record-generating institutions. To prevent this, record-generating institutions should have minimal control over the records of patients whereby empowering patients over their own records. Patients may have different attitudes in terms of sensitivity of their record attributes. The challenge today is to provide patients with a method that allows them to share the attribute values of their records efficiently and securely while maintaining the privacy preferences they desire. Patients should also be able to selectively share certain parts of their record(s) with the healthcare providers they select. Although patients may not easily understand how to share the medical parts of their record(s), empowering them would at least allow them to consult their trusted medical staff such as their Primary Care Physicians (PCPs) to decide on how and which parts of their medical record(s) are to be shared. With their consent, patients may even delegate this process to such entities to avoid situations such as being unconscious, requiring someone to make decisions for them. Lastly, patients should not be concerned with the availability nor the guaranteed secure storage of their records.

4.1 System Model

The general model of our proposed record-sharing scheme is illustrated in Fig. 1. The system consists of six main entities:

Patients. Patients are data owners that wish to share their data selectively based on their privacy preferences. We denote the set of patients as $\{p_a \in \mathcal{P} \mid 1 \leq a \leq \infty\}$.

Medical Institutions. Medical institutions provide treatment and generate medical records for the patients. We denote the set of medical institutions as $\{m_b \in \mathcal{M} \mid 1 \leq b \leq \infty\}$.

Staff Members. Personnel employed at a medical institution. We denote the set of staff members at medical institution m_b as $\{s_{b,l} \mid \forall\ 1 \leq l \leq \infty\}$. Staff members are categorized into groups of similar characteristics based on the set of attributes $\mathbb{A} = \{\mathsf{A}_1, \mathsf{A}_2 \ldots, \mathsf{A}_n\}$ they each possess. Attributes represent identifiers that are selected from an infinite pool set. They may be as general as the role of a staff member and could be as unique as a biometric.

Key-issuer. A semi-trusted party that generates keys for permissioned staff members upon their requests to access parts of the record.

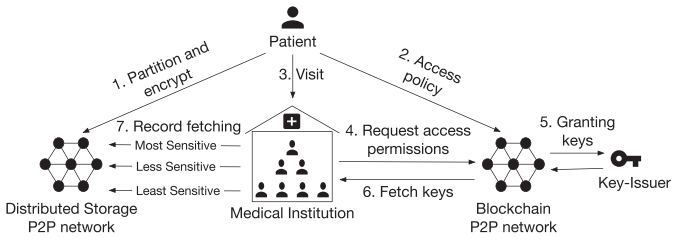

Fig. 1. General scheme orchestration.

Distributed Storage P2P Network. A non-trusted P2P network used by the patients to store and share their records with staff members at any medical institution.

Blockchain P2P Network. A non-trusted P2P network that maintains a blockchain to regulate access permissions of patient records to the staff members of medical institutions.

In general, in order for patients, medical institutions, staff members or key-issuers to interact with any of the two P2P networks, they must run nodes that are capable of connecting to either network. These nodes may be light-weight nodes (similar to the simple payment verification nodes in Bitcoin [4]) that can assemble transactions and propagate them to the network. Running light-weight clients does not require powerful computing machines and can be done via simple machines such as mobile devices making our proposed scheme accessible to everyone. The common requirement for running either node is being able to generate the public key pub and private key pr pair.

4.2 Design Goals

Our proposed scheme aims at satisfying the following:

Data Ownership. Patients are empowered over their records and control how to share them.

Fine-Grained Access Control. Patients can grant specific access permissions to the desired staff members based on their privacy preferences. They can selectively share any part of their records using any set of staff member attributes they wish, limiting access to specific parts of their record and to certain staff members at particular medical institutions.

Collusion Resistant. Two or more staff members that possess different sets of attributes and/or are employed at the same/different institution(s) cannot combine their attributes to gain access to any part of the records they are not authorized to access individually.

Data Confidentiality. Records are completely protected from any unauthorized staff members that do not possess the correct set of attributes predefined by the patient. This includes any type of space that stores the records for the patients.

Distributed Storage. Patient records are stored in a distributed storage network. They can be accessed at any time and by any permissioned staff member. Storage is not centralized and/or controlled by the record-generating institution.

Data Access Trace-Ability. Patients can trace-back to the entire history of accesses of their records. This allows patients to keep track of how their records are being accessed.

5 The Proposed d-MABE Scheme

In this section, we first present an overview of the major chain of events in our proposed scheme. For discussion

TABLE 1
Notations

Symbol	Definition
p_a	the a^{th} patient in the set of patients $\mathcal{P}$
m_b	the b^{th} medical institution in the set of institutions $\mathcal{M}$
$s_{b,l}$	the l^{th} staff member working at m_b
pr	private key of $s_{b,l}$
pub	public key of $s_{b,l}$
$\mathbb{A}$	the set of attributes of a staff member
$\mathcal{T}_i$	the i^{th} access policy defined by patient at level $\mathcal{L}_i$
sk_i	the i^{th} secret key belonging to $\mathcal{T}_i$
SBK	subscription-based key
R_i	the i^{th} partition of record $\mathcal{R}$
ER_i	the i^{th} encrypted record partition under $\mathsf{sk}_i \Vert \mathsf{SBK}$
PK	public key of key-issuer
MK	master key of key-issuer
Esk_i	encryption of sk_i under PK
SK	secret key issued for a staff member
ESK	encryption of SK under pub

purposes, we assume that a patient p_a has already received some medical treatment at institution m_b and that a medical record $\mathcal{R}$ was generated for him/her. We also assume that p_a anticipates visiting some other institution $\{m_b \mid b \geq 1\}$ in the future and would like to share $\mathcal{R}$ selectively among its staff members. Based on our description, we then propose a generalized structure through three smart contracts that can be deployed on a blockchain. We assume patients and data users interact with our smart contracts via easy-to-use and user-friendly interfaces, hence, onboarding new users to our proposed model becomes a trivial task.

The major notations used in this paper are summarized in Table 1.

5.1 Scheme Orchestration

The proposed scheme can be divided into seven major events as demonstrated in Fig. 1.

Record Partition and Encryption. Following the work presented in [12], p_a initially defines a privilege-based access structure that satisfies his/her privacy desires. The access structure is divided into k levels $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_k\}$ where each level is associated with an access policy $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$. The p_a then partitions $\mathcal{R}$ into k segments $\{\mathsf{R}_1, \mathsf{R}_2, \ldots, \mathsf{R}_k\}$ of different sensitivity. Each $\mathsf{R}_i \in \mathcal{R}$ may represent one or more record attribute(s) depending on how p_a partitions $\mathcal{R}$. Next, p_a derives a set of symmetric keys $\{\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_k\}$ as described in Section 3.2. Using these keys, p_a then encrypts each corresponding $\mathsf{R}_i \in \mathcal{R}$ as

$$\mathsf{ER}_i = \mathsf{Sym\text{-}Enc}_{\mathsf{sk}_i \Vert \mathsf{SBK}}(\mathsf{R}_i), \tag{2}$$

where Sym-Enc is a symmetric encryption algorithm such as the Advanced Encryption Algorithm (AES) and SBK denotes the subscription-based key. The SBK can only be accessed by either a system call or an attributed-based protection to active members through successful membership subscription authentication. Members have no direct access to SBK and can only use it while on-site. This design serves two purposes: (i) it prevents any users from sharing the ER_i's even if they share their sk_i's with others and give them unprivileged access and (ii) it provides an easy option to disable unsubscribed members from accessing EMRs. In other words, with this design, key revocation is no longer needed. Finally, using the Encryption function discussed in Section 3.1, p_a encrypts each sk_i under its corresponding $\mathcal{T}_i$ defined in the privilege-based access structure, such that

$$\mathsf{Esk}_i = \mathsf{CPABE\text{-}Enc}(\mathsf{PK}, \mathsf{sk}_i, \mathcal{T}_i), \tag{3}$$

where CPABE-Enc is the CP-ABE encryption function and PK is the public key generated during the Setup phase. The p_a then stores the generated ciphertexts $\{\mathsf{ER}_1, \mathsf{ER}_2, \ldots, \mathsf{ER}_k\}$ and $\{\mathsf{Esk}_1, \mathsf{Esk}_2, \ldots, \mathsf{Esk}_k\}$ over IPFS.

In cases that EMRs consist of multiple/diverse attributes, it may be a burden for patients to define their privilege-based access structures and may even leak sensitive information if defined inappropriately. A possible approach that patients may consider is to divide their records into multiple categories based on the nature of data being shared. For example, a record could be divided into personal information, medical information, and financial information. For each category, a patient can then define a separate privilege-based access structure which would reduce the overall complexity of each structure. Patients may even consult their trusted Primary Care Physicians on how to define a privilege-based access structure for proper medical information sharing.

Access Policy. To facilitate data management and sharing, p_a shares the privilege-based access structure that incorporates the access policies $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$. The access structure is incorporated into a smart contract that is then deployed over the blockchain.

Medical Institution Visit. When p_a visits a medical institution $\{m_b \mid b \geq 1\}$ to get medical treatment, p_a interacts with various staff members that may or may not belong to a predefined privilege-based access structure.

Requesting Access Permissions. Permissioned staff members will be granted access to certain parts of a record based on the attributes they possess. To request access, a staff member $s_{b,l}$ interacts with the smart contract previously deployed by p_a that incorporates the privilege-based access structure. The smart contract will verify whether the $s_{b,l}$ possesses a set of attributes that can satisfy an access policy within the access structure. Once the verification is performed the smart contract will publicly announce the access permissions of the $s_{b,l}$ over the blockchain.

Granting Keys. The key-issuer continuously monitors the blockchain for access announcements. An announcement contains the set of attributes $\mathbb{A}$ that the $s_{b,l}$ possesses. It is a form of verification to the key-issuer that the $s_{b,l}$ possesses set $\mathbb{A}$ before generating a secret key SK using the KeyGeneration function described in Section 3.1. The key-issuer then encrypts SK with the public key pub of $s_{b,l}$ as

$$\mathsf{ESK} = \mathsf{Asym\text{-}Enc}_{\mathsf{pub}}(\mathsf{SK}), \tag{4}$$

Algorithm 1. SMR Smart Contract

```
1  struct staffMember contains
2      staffID
3      staffAttributes[upBound]
4  end
5  variable mapping(address → staffMember) Map
   // Adding a new staff member
6  function
     addStaffMember(_address, _id, _attributes[upBound])
7      if (msg.sender ∉ setOfCertifiers) then
8        throw
9      else
10       Map(_address → _id, _attributes[upBound])
11     end
12 end
   // Return the attributes of a staff member
13 function getAttributes(_address)
14     return Map[_address].staffAttributes
15 end
```

Algorithm 2. AVPA Smart Contract

```
1  Initialized variables _address = SMR address
2  event LogAnnounce(address, attributes, T_i)
3  function verifyRequest(_msg, σ)
       // Verify 1: validate digital signature
4      signer ← ecrecover(_msg, σ)
5      if (signer ≠ Hash(pub)) then
6        throw
7      else
         // Verify 2: fetch stored attributes of the staff member
8        r ← SMR(_address)
          fetched ← r.getAttributes(signer)
         // Verify 2: fetched attr. versus access policies
9        for i ← 1 to k do
10         if (satisfy(T_i, fetched) == true) then
11           LogAnnounce(signer, fetched, T_i)
12           break
13         end
14       end
15     end
16 end
```

where Asym-Enc is the asymmetric encryption function. To share the key with the $s_{b,l}$, the key-issuer transacts with a global smart contract that publicly announces the encrypted key over the blockchain.

Key Fetching. The $s_{b,l}$ can obtain the uniquely encrypted secret key ESK by monitoring the announcements made over the blockchain. Once obtained, the $s_{b,l}$ can decrypt ESK using his/her private key pr that corresponds to the public key pub such that

$$\mathsf{SK} = \mathsf{Asym\text{-}Dec}_{\mathsf{pr}}(\mathsf{ESK}), \tag{5}$$

where Asym-Dec is the asymmetric decryption function.

Record Fetching. The storage location of the encrypted EMRs over IPFS is also provided to the $s_{b,l}$ in an encrypted form under his/her public key in the announcement made over the blockchain. The $s_{b,l}$ decrypts the IPFS location to fetch the EMRs. Here, the $s_{b,l}$ possesses the necessary key to decrypt the parts he/she has been granted access to. Once fetched, the $s_{b,l}$ can decrypt the encrypted symmetric key Esk_i using the derived key SK as explained in Section 3.1. That is

$$\mathsf{sk}_i = \mathsf{CPABE\text{-}Dec}(\mathsf{Esk}_i, \mathsf{SK}), \tag{6}$$

where CPABE-Dec is the CP-ABE decryption function. After obtaining sk_i, the $s_{b,l}$ can derive the remaining set of secret keys $\{\mathsf{sk}_{i+1}, \ldots, \mathsf{sk}_k\}$ using Equation (1). Finally, the $s_{b,l}$ can decrypt each encrypted partition, such that

$$\mathsf{R}_i = \mathsf{Sym\text{-}Dec}_{\mathsf{sk}_i \| \mathsf{SBK}}(\mathsf{ER}_i), \tag{7}$$

where Sym-Dec is the decryption function.

5.2 Smart Contracts

Our proposed scheme includes three main smart contracts: (i) staff member registration, (ii) access verification and permission announcements, and (iii) granting keys.

Staff Member Registration. Staff Member Registration (SMR) is a global smart contract that registers staff members of medical institutions for patient EMRs access. The process of registering staff members by interacting with this smart contract can be delegated to a number of certified institutions that verify staff members against their possessed attributes before registering them. This is achieved by incorporating precise policies into the smart contract which requires certain identities to trigger it. Once a certified institution verifies the attributes of a staff member, it executes the smart contract and uses the attributes as input. This results in the attributes are stored over the blockchain. This process maps the identity (in our case the Ethereum address) of the staff member to the set of attributes possessed. By observing the blockchain, we can trace-back the on-going activity of these certified institutions as they add, delete, or modify the information of staff members, hence, we can also detect malicious data manipulation.

Algorithm 1 outlines the general functions of the SMR smart contract. Lines 1-4 represent a generalized data structure staffMember that the smart contract uses to store new staff members. It incorporates the identity staffID and the array of attributes staffAttributes of the staff member. Lines 6-15 represent the two main functions addStaffMember and getAttributes of the smart contract. The addStaffMember function allows only certified institutions setOfCertifiers to upload the data of new staff members after physically verifying their attributes. This conditioned access is to prevent malicious attackers from granting access to themselves or others. On the contrary, the getAttributes function can be called by any user. Given the address _address of a specific staff member, this function returns the previously verified and stored attributes of this staff member.

Access Verification and Permission Announcements. The Access Verification and Permission Announcements (AVPA) is a unique smart contract defined by each patient that incorporates his/her personally defined privilege-based access structure in order to facilitate the selective record management and sharing. The staff members interested in obtaining parts of the record interact with AVPA contracts to request access permissions. The smart contract is outlined in Algorithm 2.

Algorithm 3. GK Smart Contract

```
1  event LogKeys(staffAddress, ipfsAddress)
   // Sharing the location of a generated access key
2  function addKey(_staffAddress, _ipfsStorageAddress)
3      if (msg.sender ∉ setOfIssuers) then
4          throw
5      else
6          LogKeys(_staffAddress, _ipfsStorageAddress)
7      end
8  end
```

The smart contract performs a sequential verification process. This is represented in a single function verifyRequest that takes two inputs: a message _msg prior to being signed and its signature σ. That is

$$\_msg = \mathsf{Hash}(\mathsf{staffID}), \tag{8}$$

$$\sigma = \mathsf{Sign}(\mathsf{pr}, \mathsf{pub}, \_msg), \tag{9}$$

where Hash is the hashing function and Sign is an ECDSA signing function.

In the initial verification process, the AVPA smart contract uses an ECDSA compatible validation function $\mathsf{ecrecover}(\_msg, \sigma)$ to validate σ by verifying whether

$$\mathsf{ecrecover}(\_msg, \sigma) = \mathsf{Hash}(\mathsf{pub}), \tag{10}$$

holds true. In fact, if the validation function is conducted truthfully, then the correctness of Equation (10) follows from the ECDSA. Next, the signer is compared to the address of the requesting staff member as shown in line 5. If the addresses match, the contract uses the signer to fetch the previously verified and stored attributes of this staff member in the SMR contract. However, this method is liable to replay attacks. In Section 6, we discuss this issue and our proposed countermeasure.

If the initial verification is successful, the contract executes the second verification as outlined in lines 10-14. In this process, the fetched attributes are checked against the access policy $\mathcal{T}_i$ within the access structure. The access structure is defined by the patient during contract deployment and maintains the desired privacy settings as discussed in Equations (2) and (3). The access policies are tested sequentially starting from the highest ranked access policy $\mathcal{T}_1$ at level $\mathcal{L}_1$. If the attributes satisfy the access policy $\mathcal{T}_i$, the verification is discontinued and the function fires an event LogAnnounce that announces a permanent log of the smart contract transaction stored over the blockchain. This event is a public and immutable announcement to ensure that the staff member in possession of signer should be granted access to the parts of the record $\{\mathsf{R}_i \dots R_k\}$ corresponding to levels $\{\mathcal{L}_i \dots \mathcal{L}_k\}$.

Granting Keys. Granting Keys (GK) is a global smart contract that is used to share the location of the generated and encrypted access keys over the distributed storage. The contract generally consists of a single function, addKey, as outlined in Algorithm 3.

When the key-issuer observes a LogAnnounce event fired by the AVPA smart contract, it generates an access secret key SK for the specified staff member using the unique set of attributes fetched stored in the logs. Next, the key-issuer encrypts this access key as shown in Equation (4) with the public key of the staff member. It also encrypts the IPFS record location as $\mathsf{ERA} = \mathsf{Asym\text{-}Enc}_{\mathsf{pub}}(\_ipfsRecordAddress)$ with the same public key and uploads both values to the IPFS storage, maintaining their reference location. Following that, the smart contract fires an event, LogKeys as shown in line 6, which permanently stores the address of the requesting staff member _staffAddress along with the IPFS storage location _ipfsStorageAddress. Using _ipfsStorageAddress, the staff member can fetch the encrypted key ESK and ERA stored over the network. However, as shown in the addKey function, only certified key-issuers setOfIssuers are capable of triggering this function. Therefore, attackers are prevented from spamming the smart contract logs with fake keys or addresses. The staff member can listen for these fired events and obtain the corresponding _ipfsStorageAddress as it appears in the logs. Using the _ipfsStorageAddress, the staff member can fetch ESK and ERA from the network. It is important to note that only the staff member in possession of the private key pr corresponding to the public key pub will be able to decrypt the fetched encrypted key and IPFS record address. Attackers continuously listening to the fired events may be able to learn certain IPFS addresses storing generated encrypted keys, but not the actual keys or the record address. First the data user decrypts $\_ipfsRecordAddress = \mathsf{Dec}_{\mathsf{pr}}(\mathsf{ERA})$ to learn the location of the encrypted record. Next, using the obtained secret key SK, the staff member can then decrypt Esk_i to generate the secret key sk_i. Finally, the staff member can decrypt ER_i stored over IPFS at _ipfsRecordAddress to obtain the record part R_i.

5.3 Access Permission Revocation

Attributes possessed by members may change over time due to, for example, job switch or retirement. As a result, the access rights to the patient records should be revoked to be consistent with the access policy. However, this could be challenging since it requires the attributes of the staff members to be updated periodically. While adding an expiration date [16] to each attribute when registering staff members seems to be a simple solution, it requires the patients to incorporate time constraints into their access policies.

The proposed d-MABE scheme can handle access permission revocation efficiently without requiring any extra process for the key-issuer. The staff members only need to be registered to ensure attribute-based access control. As a result, if at any point in time a staff member is unable to provide evidence of registration to the level of access claimed, future access to the patient records will be disabled. Consequently, if the staff member attempts to request records he/she is no longer entitled to access, the AVPA smart contract will fail to verify the request.

For data users that have already accessed certain records and are no longer registered, our proposed scheme can also revoke their access permissions since each record partition R_i is encrypted under $\mathsf{sk}_i \| \mathsf{SBK}$. To revoke access from those data users that are no longer registered, the patient only needs to generate a new subscription-based key SBK then re-encrypt his/her record partitions under $\mathsf{sk}_i \| \mathsf{SBK}$. This design does not require regenerating a new set of keys $\{\mathsf{sk}'_1 \dots \mathsf{sk}'_k\}$ or making any changes to the privilege-based

access structure. To prevent any data user that has been granted key sk_i at some point with an inactive membership from accessing the record partitions, we only need to re-encrypt the record partitions either periodically (such as monthly), or based on demand.

6 Security and Privacy Analysis

In this section, we present the security and privacy discussions of our proposed scheme. We assume our system runs over a blockchain that is maintained by a significant number of nodes, for example, Ethereum. In such blockchain platforms, it is very expensive to tamper with verified transactions processed into blocks. The best bet of the attackers would be to control more than half of the computational power in the network in order to perform 51 percent attacks. We also consider distributed storage networks such as IPFS that are content addressable. These networks use the hash computation of the original data when storing and referencing it over the network nodes, resulting in tamper-resistant storage. Moreover, we consider adversaries that can act as any entity within the system and can manage to connect to either the IPFS or the blockchain network. Such adversaries are capable of generating fraudulent transactions and propagating them through the blockchain network. By fraudulent transactions, we mean transacting with smart contracts in an effort to access data to which they are not granted access. They may also deploy their own smart contracts over the blockchain, request/store data over the IPFS nodes, and continuously monitor the network channels.

6.1 Security Analysis

We first discuss how our proposed scheme is secure against potential attacks. Next, we present good practices that may help improve the security of the system.

Theorem 1. *The proposed scheme is secure against replay attacks.*

Proof. The initial verification process performed by the AVPA smart contract requires the requesting staff members to submit ECDSA digital signatures σ to prove their identities. Given that all data sent over the blockchain is public, malicious attackers can easily maintain a copy of the submitted signatures and later on impersonate the honest staff members, giving them access to data they should not be able to access. To work around this issue, staff members are required to time-stamp their digital signatures before transacting with the AVPA contract, such that

$$\sigma \leftarrow \mathsf{Sign}(\mathsf{pr}, \mathsf{pub}, \mathsf{Hash}(\mathsf{staffID} \| \mathsf{T})), \tag{11}$$

where T is the time at which the signature is generated. Any submitted time-stamped signature is stored over the blockchain in order for the nodes to check if it has been submitted before. The blockchain nodes will not execute any requests associated with time-stamped signatures that have appeared previously. Therefore, to be able to perform replay attacks requires the attackers to reverse the transactions that contain an already used digital signature. The success of reversing a transaction can be modeled as a race between the honest miners and the attackers trying to generate blocks by competing to solve a hard cryptopuzzle known as Proof-of-Work [4]. The race can be modeled as a binomial random walk. It is denoted as z which represents the number of blocks generated by the honest miners with computational power p minus the number of blocks generated by the attackers with computational power $q = 1 - p$. This race can be derived as

$$z_{i+1} = \begin{cases} z_i + 1, & \text{with probability } p, \\ z_i - 1, & \text{with probability } q. \end{cases} \tag{12}$$

Using the negative binomial distribution, we can model the probability of success of the attackers. Assume the key generator waits for n blocks to be generated by the honest miners with computational power p before generating a private key for the requesting attacker. Also assume that, at that time, the attacker is able to secretly generate m blocks with computational power $q = 1 - p$, where $m = n - z - 1$. By definition, this can be modeled as the m number of blocks that the attacker can generate before the n number of blocks generated by the honest miners. Therefore, the probability of a reversing a transaction for a given value m is

$$P(m) = \binom{m+n-1}{m} \times p^n q^m. \tag{13}$$

Overall, the probability for an attacker to surpass successfully the number of blocks generated by the honest miners can be computed as

$$\begin{aligned} P_s &= \sum_{m=0}^{\infty} P(m) \times Q_{n-m-1} \\ &= 1 - \sum_{m=0}^{\infty} \binom{m+n-1}{m} \times p^n q^m \\ &\quad \times \begin{cases} 1 - \left(\frac{q}{p}\right)^{n-m}, & \text{if } q < p \text{ or } k \le n-m, \\ 1 - 1 = 0, & \text{if } q > p \text{ or } k > n-m. \end{cases} \end{aligned} \tag{14}$$

Equation (14) confirms that when $q > p$, the attacker will always succeed since $P_s = 1$. When $q < p$, the probability of success can be defined as

$$\begin{aligned} P_s &= 1 - \sum_{m=0}^{n-1} \binom{m+n-1}{k} \times p^n q^m \times \left(1 - \left(\frac{q}{p}\right)^{n-m}\right) \\ &= 1 - \sum_{m=0}^{n-1} \binom{m+n-1}{m} \times (p^n q^m - p^m q^n). \end{aligned} \tag{15}$$

Therefore, as more blocks are appended to the blockchain and $q < p$, replay attacks are infeasible. Moreover, by incorporating active registration, we can limit the pool of malicious attackers to only the staff members with the same level of access, making it even more infeasible. □

Theorem 2. *The proposed smart contracts are collusion resistant.*

Proof. Our proposed scheme requires staff members to be registered through the SMR contract before being able to request access permissions through the AVPA contracts. During registration, each member address is mapped to a

single set of attributes $\mathbb{A} = \{\mathsf{A}_1, \mathsf{A}_2 \ldots, \mathsf{A}_n\}$ after providing proof of possession to the certified institutions. The AVPA contract can only accept a single signature as input when triggered. Therefore, to perform a collusion attack, the attackers are required to regenerate a single digital signature of an already registered member that possesses the set of attributes desired. For ECDSA, a signature is generated by randomly selecting a cryptographically secure random integer d, such that

$$(x_1, y_1) = d \times G, \tag{16}$$

where (x_1, y_1) are the calculated elliptic curve points, G is the generator of the elliptic curve with a large prime order n, and $d \in_r [1, n-1]$. Next, the digital signature is calculated as two components $\sigma = (r, s)$. That is

$$r = x_1 \,(\text{mod } n), \tag{17}$$

$$s = d^{-1}(z + r \cdot pr) \,(\text{mod } n), \tag{18}$$

where z is the L_n leftmost bits of _msg such that L_n is the bit length of the group order n. If any of the values r or s are equal to zero, d is randomly selected again and Equations (16), (17), and (18) are recalculated. Since the signature components (r, s) are both derived based on the random value d, it is infeasible for the attackers to regenerate a specific signature that belongs to a certain staff member, hence, the smart contracts are collusion resistant. □

Theorem 3. *Using a content-addressable storage such as IPFS ensures that data is tamper-resistant.*

Proof. In IPFS, records are initially partitioned into n smaller segments before being stored. That is

$$\mathcal{R} = \{\mathsf{R}_1 \ldots \mathsf{R}_n\}, \tag{19}$$

where each $\mathsf{R}_i \in \mathcal{R}$ is 256 KB, by default. However, the size of a partition may also be customized by the patient as desired. Next, each $\mathsf{R}_i \in \mathcal{R}$ is hashed as

$$h_i = \mathsf{Hash}(\mathsf{R}_i), \tag{20}$$

where h_i is the resulting digest and is used to reference records stored in the network nodes through the DHT. For Hash, it is computationally infeasible to find an $\mathsf{R}'_i \neq \mathsf{R}_i$, such that

$$\mathsf{Hash}(\mathsf{R}'_i) = \mathsf{Hash}(\mathsf{R}_i). \tag{21}$$

Therefore, it is computationally infeasible for an attacker to tamper with the records of the patients. □

Theorem 4. *The proposed scheme can counter Distributed Denial of Service (DDoS) attempts.*

Proof. DDoS attacks over IPFS aim at congesting the network by sending numerous requests to the nodes storing the requested data. However, to perform such attacks, the attackers must initially identify all the nodes storing the target data from the DHT. Next, they must disrupt the service by sending these nodes a large number of requests, such that

$$\text{Number of requests} \gg \mathsf{MAX\_TRAN}, \tag{22}$$

where MAX_TRAN is the maximum number of transactions accepted by a node as defined in its configuration file.

Attackers may also try to isolate and monopolize all inbound and outbound connections and data flows of the storage nodes (referred to as an Eclipse attack [17]) from the entire network. These attacks become infeasible as the number of nodes storing the data increases since the record is replicated across the network nodes. In addition, such attacks are limited by the time t_{scheme} for a request to appear over the blockchain until the staff member fetches the data from the IPFS network nodes. Attackers must be able to perform the entire attack in time t_{attack}. That is

$$t_{attack} < t_{scheme}. \tag{23}$$

An attacker with no prior knowledge of which data will be requested by staff members has no advantage of identifying the nodes storing the data. Therefore, as the value of t_{scheme} becomes smaller and with an increased record replication, DDoS attacks become infeasible. □

6.2 Recommended Security Practices

Security may also be enhanced by adopting good development practices. In the following, we present some highly recommended techniques that developers should keep in mind while implementing the system.

Global Smart Contracts Access Control. It is necessary to ensure integrity of the data stored in the SMR and GK global contracts since they provide the necessary information required to verify AVPA contracts and grant keys to the staff members. Therefore, it is imperative to halt the attackers from manipulating the state of these contracts with fraudulent information. Attackers with access rights that can modify the SMR contract may easily register themselves with any set of attributes required to pass the verification of certain AVPA contracts. It is also feasible to perform distributed denial-of-service attacks by deleting or modifying the registration of staff members in the SMR contract or spamming the logs of the GK contract with fraudulent information. This will result in interrupting or slowing down the process of retrieving patient records. Therefore, when developing such contracts, it is important to limit the ability to modify the state of global contracts to only certified institutions. This is outlined in lines 7 and 3 of Algorithms 1 and 3, respectively. The current smart contract programming languages, for example, Solidity [18], provide specific functions to verify certain conditions and/or variable values before execution is performed. Well known examples include the require() or assert() functions that verify preliminary conditions and consume a portion or all of the gas associated with a transaction. In general, two main reasons for charging users gas to trigger contracts deployed over the blockchain are to reward the executing network nodes and make potential attacks expensive. This means that attackers must pay for each transaction they request for it to be executed by the network nodes. This approach will not prevent attackers from registering themselves into the SMR with a

set of fraudulent attributes but could help halt those trying to spam the network. Assuming an attacker has been able to modify the state of a global contract, the attacker will still have to pay for each requested transaction. To increase the security measures in this scenario, contracts may be designed to require a minimum gas amount in order to execute them. However, this countermeasure results in a trade-off between security and cost for the honest users.

External Smart Contract Calls. Our proposed scheme relies on external smart contract calls that fetch the stored attributes of registered staff members to compare them to those sent by the requesting staff members before validating if they satisfy a certain access policy. External calls can introduce several unexpected risks or errors if the smart contracts mistakenly execute malicious code. Therefore, it is important that patients cautiously implement their AVPA contracts to handle errors when externally calling the SMR global contracts during the initial verification process. In Solidity [18], external calls can be performed in two ways: low-level calls and contract calls. Low-level calls do not throw exceptions, instead they return false if the external call of another smart contract itself encounters an exception. Patients that use low-level calls must check if the returned value within the AVPA contract fails and then properly handle it. On the other hand, contract calls are more direct as they throw exceptions as encountered by the external call. From a security standpoint, it may be more secure to use contract calls especially when the patient is less experienced in handling complex scenarios.

Non-Public Blockchains. The security of our proposed scheme could also be enhanced by running the smart contracts over a consortium or private blockchain. In such a setting, miners are selected nodes within the peer-to-peer network that are known and trusted. If any of these nodes attempt to act maliciously, they are immediately identified and eliminated. A good example of such candidate nodes could be the medical institutions themselves that are interested in maintaining the system in return for efficient data access when required.

6.3 Privacy Analysis

While our proposed scheme aims at satisfying the privacy desires of patients and their records, it tends to leak knowledge about the staff members and their access patterns. Each staff member must initially become registered and is linked to a unique address before engaging with the system. Since this information can be leaked, attackers can simply monitor requests triggered by the staff members by observing the blockchain activity. However, since no information is revealed about the records of patients through the AVPA smart contracts, attackers can only track when staff members trigger requests, but not the data they have requested or the patient information.

From the perspective of patients, our proposed scheme is intentionally designed to allow them to trace-back the accesses performed by staff members to their records. This is possible by tracing the history of LogAnnounce events fired as outlined in line 12 of Algorithm 2. All fired events are permanently stored over the blockchain, allowing the patients to continuously monitor how their records are being accessed. If a patient notices irregular staff member accesses that are undesired, the patient can simply redeploy a new AVPA smart contract that defines new or more strict access policies.

Theorem 5. *Under the proposed model, the privacy of patient records location is preserved.*

Proof. First, since the user record is encrypted as described in Equation (4) before it is uploaded to IPFS, the content of the user record is protected. Second, for the current IPFS system, any user in possession of the data can reproduce its cryptographic-hash, search its corresponding location that is maintained by the DHT, and locate it within the peer-to-peer network nodes determined by the default IPFS partitioning techniques. This will also result in recursively locating any data linked to it. To ensure data privacy, we will append a random value r to the record before uploading it to IPFS. Therefore, the storage location is determined by

$$h = \mathsf{Hash}(\mathcal{R} \| r). \tag{24}$$

The randomness of r makes it infeasible for the attacker to locate the data stored over IPFS nodes. □

7 Performance and Application Analysis

In this section, we will first evaluate the performance of our proposed smart contracts. The performance measurement of smart contracts can be performed either through (i) measuring the computational complexity of the algorithms, or (ii) calculating the gas costs required to deploy/transact with the contract.

Method (i) analyzes the computational complexity based on the number of inputs. The computational complexity directly correlates to the gas costs paid by the data owners to deploy their smart contracts and by data users to execute them. However, with smart contracts, reducing computational complexity is not of significant importance since blockchain transactions are generally executed at discrete intervals. Therefore, reducing the computational complexity may only reduce the gas costs required rather than the total latency. On the other hand, method (ii) is usually more accurate since it presents an estimate of the actual monetary costs paid by the users in order to deploy/transact with the contract. However, this method is directly dependent on the actual source code implementation. Therefore, code optimization techniques may probably end up reducing more gas costs. To measure the performance of our proposed smart contracts, we will present discussions on each of our proposed algorithms that combine both methods. Our discussions assume that smart contracts will be implemented in Solidity [18], a contract-oriented, high-level language for implementing smart contracts deployable over the Ethereuem blockchain and processed by the EVM.

Following that, we analyze the symmetric encryption overhead and present our discussion for EMRs re-encryption.

7.1 Smart Contracts Computational Complexities

SMR *Contract.* As outlined in Algorithm 1, the addStaffMember function takes _attributes as input to register a new staff

member. Due to the computational limitations of the EVM, returning dynamic content from external function calls is not possible, i.e., returning a dynamically sized array of attributes resulting from the getAttributes function when called by the AVPA contract. This means, the size of Map[_address].staffAttributes must be fixed in order to be executed by the EVM. As a result, the only workaround is to use statically-sized upBound arrays of attributes as input to the addStaffMember. In this case, performance is based on the size of upBound, such that

$$\text{Performance} \propto \mathsf{upBound}. \tag{25}$$

AVPA *Contract.* As discussed previously in Algorithm 2, the AVPA contract consists of two sequential verifications that play a dominant role in the performance of the contract. In the initial verification, an input digital signature σ is validated to prove the identity of the requesting staff member. With Solidity, the only available cryptographic function that can perform such a process is the ECDSA ecrecover function. The function recovers the Ethereum address associated with the public key from the elliptic curve signature or returns zero on error. At the time of writing, the ecrecover function requires 3,000 gas units. In comparison to most of the available possible computations available by the EVM [6], the ecrecover function is considered to be relatively expensive. However, this initial verification is fixed with each AVPA contract transaction.

Assuming the initial verification is successful, the AVPA externally fetches the attributes of the staff member to test them against the incorporated access structure. Similar to the SMR contract, the performance of the AVPA contract is dependent on the upBound of the fetched attributes. Finally, the fetched attributes are tested against the access policies starting at the highest level within the hierarchy. Here, performance is affected by the complexity of the overall access structure defined and the number of levels k within the hierarchy. Assuming the worst case scenario, a requesting staff member might have his/her attributes tested against all access policies within the access structure and only receive the least sensitive part R_k of the record or nothing at all. The computational complexity can be represented as $\mathcal{O}(k \times \mathsf{upBound} \times |\mathcal{T}_i|)$, where $|\mathcal{T}_i|$ is the number of attributes that form the access policy at level $\mathcal{L}_i$. To reduce this complexity, we can modify the AVPA contract such that the staff members can request that their attributes be tested against only specific access policies rather than being tested sequentially against all policies. This would reduce the computational complexity to $\mathcal{O}(\mathsf{upBound} \times |\mathcal{T}_i|)$. We can also incorporate optimized search functions, for example, binary search, that can help optimize searching for attributes in the fetched set against those in the access policies. This means that we can further reduce the computational complexity to $\mathcal{O}(\mathsf{upBound} \times \log|\mathcal{T}_i|)$. However, it is important to note that in such a scenario, the patients need to make their access structures publicly available in order for the staff members to request certain access policies. This results in a trade-off between the performance of the AVPA contract and the privacy of the access policy defined.

GK *Contract.* The GK contract requires the least computational complexity and gas costs when compared to the SMR and AVPA smart contracts. The computations involved are as simple as firing a LogKeys event when access permissions are granted to staff members. The gas costs of such operations are minimal in comparison to the other computations in the SMR and AVPA smart contracts.

7.2 Smart Contracts Gas Costs

In this section, we implement, simulate and present the estimated costs required to deploy/transact with the smart contracts of our proposed scheme in multiple scenarios. Our experiments are performed over the Ethereum testnet blockchain [19]. We specifically choose the Ethereum blockchain given that it is by far the most widely used smart contract hosting public blockchain. We intend to show that costs of running d-MABE over a public blockchain are reasonable and that our results may even be improved when choosing a consortium or private blockchain. We also note that our proposed scheme can be implemented over any other blockchain that incorporates smart contracts.

In our simulation, the AVPA smart contract with the highest degree of complexity is implemented. When executed, the AVPA smart contracts sequentially check whether the attributes of the requesting users satisfy the access policies in order starting from the highest level of the hierarchy. As discussed in Section 7, there is a trade-off between privacy and performance. Therefore, we note that our presented results can be further enhanced in terms of performance as discussed previously. However, we intentionally choose to implement our smart contracts this way to show that even with these conditions, our proposed contracts are still cheaper when compared to the current systems used by the record-generating institutions to share medical records. In contrast to the current record-sharing systems, our proposed scheme eliminates the role of the record-generating institution completely. This results in eliminating other overhead costs required when sharing medical records. For example, in developed countries such as the U.S., record-generating institutions must comply with certain state laws that enforce a maximum fee a record-generating institution may charge when requested to share its records. Table 3 illustrates a sample of these fees that could be inclusive or exclusive to the methods of delivering the record itself. Given the current competitiveness, in most cases, the medical institutions would charge the entire allowed fees to maximize their profits.

For our simulations, we apply the values $k = \{5, 10\}$, $\mathsf{upBound} = \{10, 20, 30, 40, 50\}$ and $N = \{25, 50, 100\}$, where k is the number of levels in the hierarchy and N is the total number of attributes used in the entire access structure. Without loss of generality, we also assume that the number of attributes for each level follows the uniform distribution. For example, if $k = 5$ and $N = 50$, then the number of attributes used to define an access policy is $|\mathcal{T}_i| = 10$. The following experiments have been conducted through an Ethereum node running on a system with 1.8 GHz Intel Core i5 and 8 GB RAM. Our numerical results are also the averages of 10 trials under each scenario.

Based on our experiments, the costs of deploying our smart contracts are not affected by the value of upBound

TABLE 2
Estimated Smart Contract Deployment Costs

Smart	$k=5, N=25$				$k=5, N=50$				$k=10, N=100$			
Contract	Gas Limit	Cost/ETH	Cost/USD	Data/bytes	Gas Limit	Cost/ETH	Cost/USD	Data/bytes	Gas Limit	Cost/ETH	Cost/USD	Data/bytes
SMR	240,383	0.004809	1.4	736	240,447	0.004809	1.42	736	240,447	0.004809	1.42	736
AVPA	857,419	0.017148	5.21	3,323	1,132,628	0.022653	6.67	4,536	1,303,607	0.026072	7.56	5,188
GK	125,617	0.002512	0.74	304	125,617	0.002512	0.74	304	125,617	0.002512	0.74	304

since there is no major change in code as this value changes. Therefore, for all values of upBound that we tested, the costs remained constant. Table 2 summarizes these costs in terms of gas limits and their estimated and equivalent costs in ETH and United States Dollar (USD) at the time of testing. To measure the optimum gas limit for each smart contract deployment, we used the JSON-RPC method, eth_estimateGas [20], that generates and returns an estimate of the required gas limit based on the network success rate. We also set the gas price to 20 GWEI, where 1 ETH = 1×10^9 GWEI. We intentionally select this value relatively higher than the average gas prices specified in [21] for guaranteed and fast processing. Again, we note that our presented results can be further optimized by selecting reduced gas price values. As demonstrated in Table 2, the estimated costs for deploying the SMR and GK smart contracts remain constant as we alter the values of k and N. However, the costs for deploying the AVPA smart contracts increase with an increase in the values k and/or N.

Fig. 2 demonstrates the changes in USD costs of transacting with already deployed AVPA contracts as we alter the values k and N for permissioned staff members at levels $\mathcal{L}_1$, $\mathcal{L}_5$ and $\mathcal{L}_{10}$. As shown by the figure, we recognize an increase in costs as these values increase. We also realize that as the upBound value increases while keeping the values k, N, and $\mathcal{L}_i$ constant, the costs slightly increase.

Finally, in Tables 4 and 5, we present the gas limits and costs in both ETH and USD to transact with the addStaffMember and addKey functions. As shown in Table 4, the costs of the transactions increase as the value upBound increases. On the contrary, as presented in Table 5, the costs of transacting with the addKey function remains constant regardless of the upBound value used.

TABLE 3
U.S Maximum State Fees [22] to Copy and Deliver Records Upon Being Requested by Others

State	Search fee	Cost per page	Misc. costs per page
California	N/A	\$0.25	Microfilm: \$0.50
Texas	\$82.95	P.1-10: \$45.79 flat fee P.11-60: \$1.54 P.61-400: \$0.76 P.401+: \$0.41	Microfilm: P.1-10: \$69.74 flat fee P.11+: \$1.54
Florida	\$1.00/year	\$1.00	Microfilm: \$2.00
New York	N/A	\$0.75	X-rays: Actual cost of reproduction
Illinois	\$27.91	P.1-25: \$1.05 P.26-50: \$0.70 P.50+: \$0.35	Microfilm: \$1.74

7.3 Symmetric Encryption Overhead

Our proposed work does not require patients to reconstruct their privilege-based access trees and regenerate a new set of keys. In order to revoke access from certain data users, patients need only to symmetrically re-encrypt their EMRs under the same set of keys $\{\mathsf{sk}_1, \ldots, \mathsf{sk}_k\}$ along with a new SBK. The overhead for re-encrypting EMRs can be very low when utilizing a high encryption/decryption implementation such as Advanced Encryption Standard (AES), Counter Mode (CTR) operations, and Cipher-Block-Chaining (CBC).

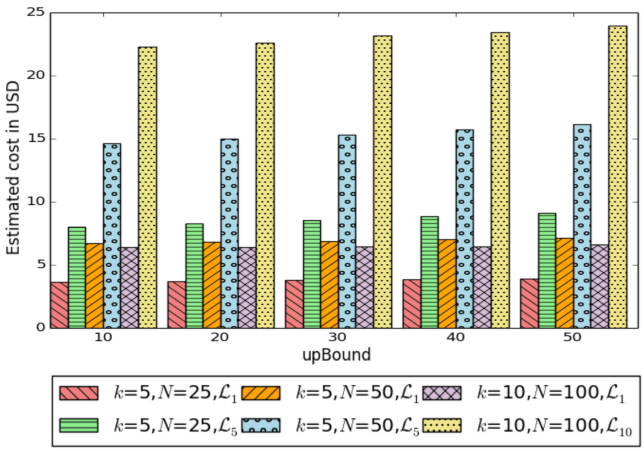

Fig. 2. Estimated costs to run the AVPA smart contract.

TABLE 4
Estimated addStaffMember Function Costs

	upBound				
	10	20	30	40	50
Gas Limit	246,531	449,890	653,249	856,674	1,060,034
Cost/ETH	0.004931	0.008998	0.013065	0.017133	0.021201
Cost/USD	1.45	2.65	3.85	5.06	6.29

TABLE 5
Estimated addKey Function Costs

	upBound				
	10	20	30	40	50
Gas Limit	26,379	26,380	26,375	26,379	26,378
Cost/ETH	0.000528	0.000528	0.00053	0.000524	0.000528
Cost/USD	0.14	0.14	0.14	0.15	0.15

According to the Crypto++ 5.6.0 Benchmarks,[1] the performance for AES/CTR on an Intel Core 2 1.83 GHz processor measures under Windows Vista are 139 MiB/Second, 113 MiB/Second, and 96 MiB/Second for 128-bit, 192-bit, and 256-bit key sizes respectively. Similarly, the performance measures for AES/CBC are 109 MiB/Second, 92 MiB/Second, and 80 MiB/Second for 128-bit, 192-bit, and 256-bit key sizes respectively. These commonly used cryptographic algorithms are widely accepted and incur minimal costs making our model feasible.

7.4 Extended Application Discussions

Aggregated patient data has the capability of transforming the future standard of care for patients through the application of predictive analytics and big data methods. One of the biggest challenges to using health data to its fullest extent is the inaccessibility and segmentation of EMRs [23]. As demonstrated by our analyses, the proposed scheme can eliminate these constraints. It grants healthcare providers the ability to efficiently extract knowledge from disconnected EMRs that have been willingly shared. For example, a meaningful opportunity for big data analytics in this context is the capability to model drug and treatment efficacy. Healthcare providers can access previously shared EMRs to understand how life-saving drugs and treatments perform, respective to the disease state and profile of the patient. This data can be used to make enhanced and customized treatment decisions for patients. As healthcare treatment becomes more individualized, successful patient outcomes are more likely and a one-size-fits-all approach to patient treatment is no longer adequate.

8 Conclusion

In this paper, we proposed d-MABE, a secure, privacy-preserving and distributed data sharing scheme that runs over a blockchain using smart contracts. We demonstrated that d-MABE can be used in cases such as medical record-sharing where patients require an efficient and selective method to share their records with staff members of medical institutions. Our d-MABE proposed scheme eliminates the reliance on the record-generating institutions when data is shared and completely empowers the patients. Our security and privacy analyses show that d-MABE is secure and preserves the privacy of patient records. We also presented some recommended security practices developers should keep in mind when developing their system. Finally, our comprehensive evaluation proves the efficiency of d-MABE and presents numerical results that support our performance analysis.

Acknowledgments

The authors would like to thank the associate editor and the anonymous reviewers for their valuable comments. This project was supported in part by NSF Grants CCF-1919154 and ECCS-1923409.

1. https://www.cryptopp.com/benchmarks.html

References

[1] U.S. Dept of Health and Human Services, "Heath insurance portability and accountability act," 2018. [Online]. Available: https://www.hhs.gov/hipaa
[2] N. England, "NHS england and NHS improvement," 2019. [Online]. Available: https://www.england.nhs.uk
[3] N. Digital, "Summary care records (SCR)-information for patients," 2019. [Online]. Available: https://digital.nhs.uk/
[4] S. Nakamoto, "Bitcoin: A P2P electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[5] N. Szabo, "Smart contracts: Building blocks for digital markets," *EXTROPY: The J. Transhumanist Thought, (16)*, vol. 18, no. 2, 1996.
[6] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.
[7] E. Zaghloul, T. Li, and J. Ren, "An attribute-based distributed data sharing scheme," in *Proc. IEEE Global Commun.*, 2018, pp. 1–6.
[8] G. Zyskind, O. Nathan, and A. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *Proc. IEEE Security Privacy Workshops*, 2015, pp. 180–184.
[9] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proc. 2nd Int. Conf. Open Big Data*, 2016, pp. 25–30.
[10] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, "Secure and trustable electronic medical records sharing using blockchain," in *Proc. AMIA Annu. Symp. Proc.*, 2017, pp. 650–659.
[11] H. T. Vo, A. Kundu, and M. K. Mohania, "Research directions in blockchain data management and analytics," in *Proc. 21st Int. Conf. Extending Database Technol.*, 2018, pp. 445–448.
[12] E. Zaghloul, K. Zhou, and J. Ren, "P-MOD: Secure privilege-based multilevel organizational data-sharing in cloud computing," *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2019.2907133.
[13] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform, 2013," 2017. [Online]. Available: http://ethereum.org/ethereum.html
[14] J. Benet, "IPFS - Content addressed, versioned, P2P file system (draft 3)," Jul. 25, 2019. [Online]. Available: https://github.com/ipfs/papers/raw/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf
[15] Bittorrent, 2018. [Online]. Available: http://www.bittorrent.com
[16] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," *J. Comput. Secur.*, vol. 18, no. 5, pp. 799–837, 2010.
[17] A. Singh, T. wan Ngan, P. Druschel, and D. S. Wallach, "Eclipse attacks on overlay networks: Threats and defenses," in *Proc. 25th IEEE Int. Conf. Comput. Commun.*, 2006, pp. 1–12.
[18] Solidity, 2018. [Online]. Available: http://solidity.readthedocs.io/en/v0.4.24/
[19] Ropsten (revival) testnet, 2018. [Online]. Available: https://ropsten.etherscan.io
[20] Json RPC, 2018. [Online]. Available: https://github.com/ethereum/wiki/wiki/JSON-RPC#eth_estimategas
[21] Ethereum average gas price chart, 2019. [Online]. Available: http://etherscan.io/chart/gasprice
[22] MediCopy, "State-by-state guide of medical record copying fees," 2018. [Online]. Available: https://medicopy.net
[23] S. E. White, "A review of big data in health care: Challenges and opportunities," *Open Access Bioinf.*, vol. 6, pp. 13–18, 2014.

Ehab Zaghloul received the BS and MSc degrees in computer engineering from Arab Academy for Science and Technology (AAST), Alexandria, Egypt, in 2012 and 2015 respectively, and the PhD degree in electrical and computer engineering from Michigan State University (MSU), East Lansing, Michigan. His research interests include applied cryptography, secure and private cloud data sharing, cryptocurrencies, and blockchain.

Tongtong Li (Senior Member, IEEE) received the PhD degree in electrical engineering from Auburn University, Auburn, Alabama, in 2000. From 2000 to 2002, she was with Bell Labs, and had been working on the design and implementation of 3G and 4G systems. Since 2002, she has been with Michigan State University, East Lansing, Michigan, where she is currently an associate professor. Her research interests fall into the areas of wireless and wired communications, wireless security, information theory and statistical signal processing, with applications in neuroscience. She is a recipient of a National Science Foundation (NSF) Career Award (2008) for her research on efficient and reliable wireless communications. She served as an associate editor for the *IEEE Signal Processing Letters* from 2007-2009, and an editorial board member for the *EURASIP Journal Wireless Communications and Networking* from 2004-2011. She served as an associate editor for the *IEEE Transactions on Signal Processing* from 2012-2016.

Matt W. Mutka (Fellow, IEEE) received the BS degree in electrical engineering from the University of Missouri-Rolla, Rolla, Missouri, the MS degree in electrical engineering from Stanford University, Stanford, California, and the PhD degree in computer sciences from the University of Wisconsin-Madison, Madison, Wisconsin. He is currently a professor with the faculty of the Department of Computer Science and Engineering, Michigan State University (MSU), East Lansing, Michigan and is currently serving as a program director at the National Science Foundation (NSF) within the Division of Computer and Networks Systems (CNS) of the Directorate for Computer and Information Science and Engineering (CISE). He served as chairperson of the MSU Department of Computer Science and Engineering from 2007-2017. He has been a visiting scholar at the University of Helsinki, Helsinki, Finland, and a member of technical staff at Bell Laboratories in Denver, Colorado. He was honored with the MSU Distinguished Faculty Award. His current research interests include mobile computing, sensor networking and wireless networking.

Jian Ren (Senior Member, IEEE) received the BS and MS degrees both in mathematics from Shaanxi Normal University, China, and the PhD degree in EE from Xidian University, China. He is currently an associate professor with the Department of ECE, Michigan State University, East Lansing, Michigan. His current research interests include network security, cloud computing security, privacy-preserving communications, distributed network storage, and Internet of Things. He is a recipient of the US National Science Foundation Faculty Early Career Development (CAREER) Award, in 2009. He served as the TPC chair of IEEE ICNC'17, general chair of ICNC'18, and executive chair of ICNC'19 and ICNC'20. Currently he serves as an associate editor for the *IEEE Transactions on Mobile Computing (TMC)*, *IEEE Internet of Things Journal (IoT)*, *ACM Transactions on Sensor Networks (TOSN)* and a senior associate editor for the *IET Communications*.