Cryptographically Enforced Access Control in Blockchain-Based Platforms

Shadan Ghaffaripour and Ali Miri
Computer Science Department
Ryerson University
Toronto, Canada
Email: s4ghaffa@scs.ryerson.ca, samiri@scs.ryerson.ca

***Abstract*—In this paper, we took a two-stage approach to address privacy issues in blockchain-based applications. Using smart contracts, our approach automatically enforces access policies through encryption and is in contrast to most access control models that solely rely on smart contract policies. These contracts are prone to many security vulnerabilities and in some cases written by developers whose trustworthiness is dubious. A cryptically enforced access control, on the other hand, gives more of the sovereignty promised by the blockchain technology, back to users as showcased in our medical data management framework.**

***Keywords*-Blockchain; Multi-authority Attribute-based Encryption; Access Control;**

I. Introduction

The design and development of privacy-sensitive applications have always been a great challenge. These applications, which handle highly-sensitive personal data must ensure their users' privacy by having an efficient access control mechanism in place.

Personal Health Records (PHR) are among the most sensitive information that such applications need to manage. It comes as no surprise that the number one concern surrounding medical data management systems, that are used as the case study of this paper, is privacy. For years, Cloud platforms were deemed a natural fit to host PHRs for medical data management systems. Even though Cloud is a cost-effective and efficient solution for this matter, it lacks an important ingredient; It does not give patients full control over their own data. To date, there have been many incidents of privacy invasions, as users placed more trust in Cloud providers than they should have.

In 2008, Bitcoin [1] and the underlying technology, blockchain, offered a glimmer of hope for the future of web applications. Today, Blockchain-based applications have a more decentralized approach; power and control are not in the hands of a single party, but instead, are distributed among all participants. The promise of the technology is to bring back the element of trust by guaranteeing transparency over how the application works; transparency over all the activities within the system, all through an immutable public ledger.

The original design of Bitcoin did not take privacy issues into account. In fact, this was not required, as it was a public, permission-less system with pseudo anonymous actors. With that said, in order to cover broader use cases, the topic of privacy in blockchain has received more attention recently. This paper seeks to address privacy in blockchain-based applications from a new perspective.

The application of the proposed framework can be extended to any similar scenario such as the medical data management system, which we have showcased throughout this paper.

A. Organization

This paper is organized as follows: Section II gives a brief overview of relevant work. Next, background information on the blockchain, smart contract, and attribute-based encryption are given in section III. A new framework is outlined in section IV, followed by evaluation of our framework in section V. Conclusions and future work can be found in section VI.

II. Related Work

It is only since the seminal paper of [1] that the study of blockchain technology has gained a huge amount of attention. In recent years, there has been an increasing amount of literature on the potential blockchain application beyond cryptocurrencies.To date, a number of these studies have focused mainly on use cases where the privacy of data on the blockchain is of critical importance.

In [2], authors presented a protocol in which users own and control their personal data by turning blockchain into an automated access control manager. One limitation of this work is that user-specified policies are defined simply as the type of data that the provider can access. For example, a policy between user u and service provider s could be defined as $policy_{s,u} = \{location, contacts\}$. With that, if the service provider is not fully trusted, it can label the data differently and access whatever data is desired.

A similar approach, as conducted in [3], developed an identity-based and role-based access control ecosystem with blockchain. The implementation of which is done in hyper-ledger fabric framework. A key assumption of this approach is that use case applies to a single organization, to whom all the data belongs and it is also the one who assigns and

978-1-7281-5052-9/19/$31.00 ©2019 IEEE

verifies roles to its limited number of users. The underlying assumption impedes a much wider adoption of their prototype to the cases where there are several data owners who, not knowing all the participants in the system, cannot easily assign roles to each and every one of them. Our approach, on the other hand, takes these cases into account and provides a generic framework.

Ekblaw et. al. in [4] highlight the importance of a system that prioritizes patients' agency and gives them a holistic view of their medical data across various organizations. Based on which they proposed a decentralized electronic health record management system, using blockchain. The given design is very comprehensive from a software engineering and a practical point of view. In particular, it is modular, abstracts the communication with blockchain, and integrates well with providers' existing local data storage. However, the mere fact that the data resides on providers local storage seems contradictory to the objective of giving patients full control over their data. Furthermore, their access control mechanism, implemented in smart contracts, ties each access query to a single public key. This mechanism would cause inefficiencies for certain scenarios. For example, consider the case where the patient wishes to share the data with all the emergency doctors throughout a given region. It is obvious that the setup in [4] would not allow the system to scale well. The Attribute-based Encryption (ABE) scheme used in this paper, is a natural fit for the mentioned scenario and can automatically enforce privacy policies to a much broader audience. This way, the complexity of the system would be linear to the number of attributes rather than the number of authorized users.

The blockchain-enabled capability-based access control framework in [5] has been specifically proposed for the security of IoTs and focuses mainly on scalability issues. For that reason, access right authorization is delegated to domain owners, in control of a subset of IoTs. Their access control mechanism uses smart contracts to implement a policy-based decision-making service based on capability tokens. With such authorization domains, the solution does not seem to be entirely decentralized. Furthermore, the solution is not designed for one-on-one transactions between two individuals as required by many systems. Our healthcare data management framework, provides a solution to this problem. The main inspiration for this research is drawn from the work of Ming et. al [6]. Furthermore, the Central-authority-less Multi-authority Attribute-based Encryption (CA-less MA-ABE) in [7] introduced a theoretical model that sets the basis of our framework.The blockchain technology is another centerpiece in this proposed solution. In what follows, required background knowledge on the blockchain, smart contact, and attribute-based encryption are provided.

III. Background

A. Blockchain and Smart Contracts

Broadly, a blockchain is an immutable ledger of transactions, maintained by a distributed network of nodes, each of which has a copy of that ledger. In this network, the state of the system can change through validated transactions that most participating nodes had agreed on. These transactions are grouped into blocks that include a hash which binds each block to the previous one. This eventually forms a chain of blocks. The main advantages of blockchain technology are transparency through a tamperproof log of transactions; decentralized governance; data integrity; automation and disintermediation.

A smart contract is a distributed application through which the internal logic of a blockchain-based application is expressed. The terms of such a contract are enforceable through the automatic execution of its code, without the intervention of any trusted intermediary. Once a smart contract has been developed and deployed on the blockchain, it cannot be changed any further due to the immutability of the blockchain.

Although smart contracts are essential building blocks of a decentralized blockchain-based application, they are also prone to serious security issues. As stated in [8], there are numerous categories of vulnerabilities that can impair the security of smart contracts and in turn, the underneath system. Therefore, in implementing access control mechanisms for such systems, relying solely on smart contracts does not give an absolute privacy guarantee, which might be of vital importance.

Even though smart contracts open up great opportunities to implement more flexible and complex access control mechanisms, it will be more advantageous if they are used in combination with cryptographic schemes.

B. Central-authority-less Multi-authority Attribute-based Encryption

Attribute-based encryption (ABE) determines whether a user can decrypt a message or not, based on their attributes [9]. In this scheme, a message is encrypted with a set of attributes, and only a user who has the minimum number of those attributes can decrypt that message. One of the possible limitations of ABE stems from the fact that a single trusted party is responsible for monitoring all the attributes and in turn, issuing decryption keys.

Multiple-authority ABE (MA-ABE) is a more suitable approach to the setting of interests in this paper. In MA-ABE, a multitude of authorities are responsible for generating keys for legit users, who hold specific attributes. Each attribute authority (AA) manages a subset of attributes in the system and therefore, can only issue keys pertaining to those attributes. The presence of multiple authorities in MA-ABE schemes does not provide an effective mechanism

for distribution of power due to the existence of a central trusted attribute authority who has the ability to decrypt any message in the system.[10].
Chase et. al. in [7] demonstrated how the central trusted authority and hence, the key escrow problem could be removed.
We believe that this scheme aligns well with the requirements of an access control mechanism in a distributed blockchain-based application that manages storage and sharing of the sensitive data belonging to multiple owners.
Below, is the summary of the technical underpinnings of our approach.

C. CA-less MA-ABE Algorithms:

This section briefly describes the main algorithms, i.e. Setup, Key Issuing, Encryption and Decryption in CA-less MA-ABE. For detailed description, please refer to [7].

Summary of Main Notations:

- g_1 and g_2 are respectively the generators of two cyclic multiplicative groups of prime order q, $\mathbb{G}_1$ and $\mathbb{G}_2$
- $\hat{e}$ is a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$
- $\mathbb{A}_k^C$ is the attribute set of a ciphertext C, in the domain of the attribute authority k
- $\mathbb{A}_k^u$ is the attribute set of a user u, handled by the attribute authority k
- Y is the master public key
- $T_{k,i}$ is the public information about the attribute i in the domain of the attribute authority k, and part of the system parameters
- $t_{k,i}$ is the private information about the attribute i in the domain of the attribute authority k; for each attribute $i \in \mathbb{A}_k$ there is a corresponding $t_{k,i}$
- $p_k(.)$ is a polynomial chosen by the attribute authority k with a degree d_k

1) Setup: This algorithm outputs public key/secret key components for each AA_k:
Master Secret Key:
Each AA_k picks $v_k \in_R \mathbb{Z}_q$ and shares $Y_k = \hat{e}(g_1, g_2)^{v_k}$ with other attribute authorities. The combination of v_ks serve as the master secret key.
The secret key components of each AA_k include:
$\langle x_k, \{s_{kj}\}_{j\in\{1,...,N\}\setminus\{k\}}, \{t_{k,i}\}_{i\in[1,...,n_k]}\rangle$
Note that s_{kj} is the pseudorandom function (PRF) seed shared between each pair of authorities k and j; x_k is the PRF base, a random number picked by authority k. Jointly, s_{kj} and x_k define the pseudorandom function $PRF(.)_{kj}$, which is computable excusively by attribute authorities k and j.
Master Public Key:
All attribute authorities compute $Y = \Pi Y_k$ independently, which accounts for the master public key of the system. Other system parameters are:
$\langle Y=\hat{e}(g_1, g_2)^{\sum_k v_k}, \{y_k=g_1^{x_k}, \{T_{k,i}=g_2^{t_{k,i}}\}_{i\in[1,..,N]}\}_{k\in[1,..,N]}\rangle$

2) Key Issuing: :
The user invokes the anonymous key issuing protocol $N-1$ times; i.e. once on each AA_k. Note here $j \in \{1, 2, ..., N\}\setminus\{k\}$. The user obtains:

$$D_{kj} = \begin{cases} g_1^{R_{kj}}.PRF_{kj}(u), \ if \ k > j \\ \frac{g_1^{R_{kj}}}{PRF_{kj}(u)}, \ if \ k < j \end{cases} \quad (1)$$

Also note that $R_{kj} \in_R \mathbb{Z}_q$ and D_{kj} values are user-specific and contained in the user secret key. Furthermore, the PRF terms will be cancelled out (in equation 3) only by the other PRF values for the same user. That is why this scheme is considered collusion resistant. Then, each AA_k picks a degree d_k polynomial, $p_k(.)$ with $p_k(0) = v_k - \sum_{j\in\{1,...,N\}\setminus\{k\}} R_{kj}$ and issues the following for each eligible attribute for the user:

$$S_{k,i} = g_1^{\frac{p(i)}{t_{k,i}}} \quad (2)$$

The user finally calculates:

$$D_u = \Pi_{(k,j)\in\{1,...,N\}\times(\{1,...,N\}\setminus\{k\})} D_{kj} = g_1^{R_u} \quad (3)$$

where,

$$R_u = \Sigma_{(k,j)\in\{1,...,N\}\times(\{1,...,N\}\setminus\{k\})} R_{kj} \quad (4)$$

Note that D_u is necessary in the decryption part to retrieve the blinding term, Y^s in the ciphertext.Please see equation 9.

3) Encryption: :
To encrypt message m for attribute set $\{\mathbb{A}_1^C, ..., \mathbb{A}_N^C\}$, user picks $s \in_R \mathbb{Z}_q$, and returns:

$$C = \langle E_0 = m.Y^s, E_1 = g_2^s, \{C_{k,i} = T_{k,i}^s\}_{i\in\mathbb{A}_k^C, \forall k\in[1..N]}\rangle \quad (5)$$

4) Decryption: :
The user does the following process for decryption:

- For each authority $k \in [1..N]$:
  for any d_k attributes $i \in \mathbb{A}_k^C \cap \mathbb{A}_k^u$, pairs up $S_{k,i}$ and $C_{k,i}$. Essentially computes:
  $$\hat{e}(S_{k,i}, C_{k,i}) = \hat{e}(g_1, g_2)^{s.p_k(i)} \quad (6)$$
- interpolates all the values $\hat{e}(g_1, g_2)^{s.p_k(i)}$ to get a function of $p_k(0)$:
  $$P_k = \hat{e}(g_1, g_2)^{s.p_k(0)} = \hat{e}(g_1, g_2)^{s.(v_k - \sum_{j\neq k} R_{kj})} \quad (7)$$
- multiplies P_ks together to get:
  $$Q = \hat{e}(g_1, g_2)^{s.(\sum\{v_k\} - R_u)} \quad (8)$$
- computes:
  $$\hat{e}(D_u, E_1).Q = \hat{e}(g_1^{R_u}, g_2^s).Q = Y^s \quad (9)$$
- recovers m by E_0/Y^s

IV. Cryptographically Enforced Access Control in Blockchain Platforms

A. Design Overview

In our proposed blockchain-powered medical data management system, we took advantage of attribute-based encryption (ABE), which allows for flexible policy-based access controls, enforceable through cryptography.

Broadly, patients establish their desired PHR access policies, based on specific attributes of the target audience. These policies are then enforced with the help of attribute-based cryptography and smart contracts, that are deployed on the blockchain. Attention should be drawn to the fact that in theory, an attribute can be anything verifiable through authorities, such as the professional role, location, and education degree in our example.

In more technical terms, what happens is that through attribute-based encryption, the PHR gets associated with a policy written across attributes governed by different authorities in various domains known as attribute authorities. Only users whose attributes match that of the PHR and in effect, satisfy that policy can decrypt the PHR using the set of keys issued by attribute authorities. In our specific example of a medical data management system, these authorities are considered to be in domains such as healthcare, education, and insurance. However, in more general terms, they are not restricted to any specific domain.

With the help of Blockchain technology, the framework achieves a decentralized and distributed mechanism for guaranteeing that the user access policies are pursued in an environment where trust is an issue.

B. Transaction Flow

In what follows, the transactional mechanics that take place during the typical *Store*, *Load* and *SecureCompute* operations are described. We suggest readers to see the transaction flow in Hyperledger fabric, given in Figure 1.

Before delving into the details of each process, it is necessary to establish the elements of the proposed medical data management system:

1) Distributed Ledger, L
2) Distributed Hash Table, DHT
3) Network Peers, i.e. participants in the medical data management platform
    a) Patients
    b) Providers, e.g. Doctors, Nurses, etc.
    c) Attribute Authorities-in various domains: Medical Domain, Educational Domain, Insurance, etc.
    d) Other Stakeholders, e.g. Researchers, etc

As a final note, we used Hyperledger Fabric[11] architecture as our reference model to describe the process. However, our framework is generic and not restricted to any specific implementation.

1) Storing PHR: :

With every data submission, the data owner, i.e. patient, provides the following inputs to the medical management application: user identifier, her/his public key, record type, identity of the record writer, access policy and encrypted content. (See Algorithm 2 line 1.) With that, a tx_{write} transaction proposal is created, signed and sent to the endorsing peers. The endorsing peers check the format of the proposal as well as the validity of the signature. The relevant smart contract, as specified by the proposal is executed on the endorsing peers, the result of which is signed and sent back to the patient as a proposal response. If the responses received, satisfy the endorsement policy, the transaction will be sent to the ordering service. The ordering service, eventually establishes the total ordering of received transactions and broadcasts them to peers of the network atomically. Once a block is delivered to a peer, the transactions within that block are validated and are tagged either valid or invalid. At this stage, the transaction, along with the hash of the PHR is stored on the peers' ledger permanently.(See Algorithm 2 line 3.) The possibly large content of the PHR, on the other hand, is recorded on a distributed hash table.(See Algorithm 2 line 5 and Figure2.) The implication of the former task is twofold; Firstly, since anything stored on the ledger is immutable, it serves as a countermeasure against manipulation of hash. As a result, with the manipulation of the data itself, the inconsistency of both data pieces will not go unnoticed. Secondly, with the transaction being permanently secured in the ledger, patients can track their health information submissions with absolute confidence about their integrity and validity. This is particularly important with respect to the access policy.Patients need to be assured that their access policy can never be tampered with.

2) Key Request: :

This process is independent of the blockchain network and can be initiated by users at any time before a data request, as a result of which users will anonymously receive the decryption keys pertaining to every attribute they hold. The details are described in section III-C2.Of note is that the key issuing protocol preserves the privacy of users who make data requests to the system; The reason is that their communications with attribute authorities are via pseudonyms. Consequently, attribute authorities cannot link multiple attributes belonging to the same user[7]. (See section III-C.)

In addition to the decryption keys, each attribute authority will sign and send a response, representing the number of attributes held by the user in its governing domain.

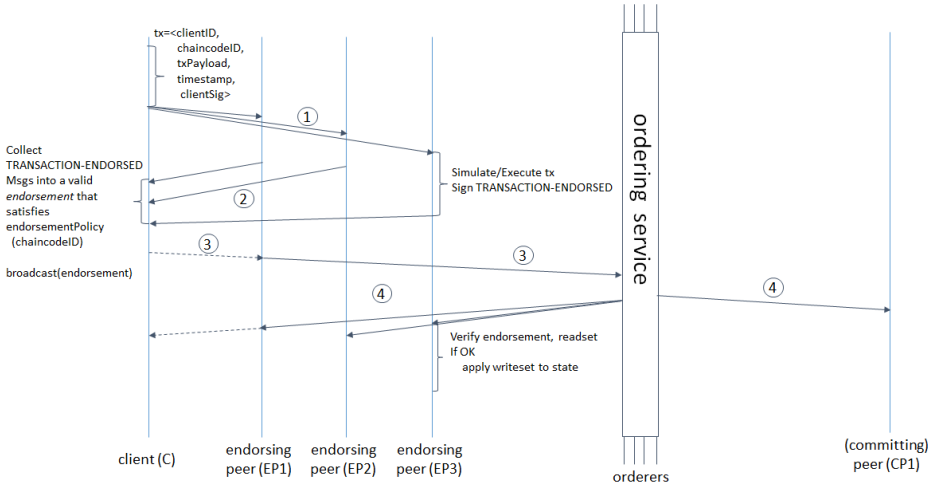

Figure 1. Transaction Flow in Hyperledger Fabric [11]

3) Loading PHR: :

With every data access request, initiated by a user, a tx_{read} transaction is generated, as a result of which a certain chaincode, indicated by the transaction, is run on endorsing peers. The code will perform the checks and send the response to the requesting user, indicating whether her roles match the access policy. (See Algorithm 3 line 4.) The tx_{read} is then passed to the ordering service with a successful status code or with an unsuccessful code otherwise.

Similar to the previous case, transactions are ordered and once validated, are stored into the ledger. In the case of a successful transaction, the requested data will also be transferred to the user. (See Algorithm 3 line 5 and Figure2.)

The importance of persisting this transaction should not be overlooked. This is where our system further assures patients that their data is never shared with unauthorized entities, not even in its encrypted format. In such a setting, the patients can take a further step and check for themselves who has retrieved/attempted to retrieve their PHR. From a business point of view, the value of this trust is staggering;users will only use a system if they fully trust it. Also, note that the use of blockchain has removed intermediation and has enabled users to securely share their healthcare data with one another without the intervention of third parties.

4) Secure Computation over PHR: :

Our framework supports two different levels of security; aside from the scenario that has been described above, certain roles should not have access to raw data. However, they still can run computations over the encrypted data. When such a request is initiated as a $tx_{compute}$, the system will first check to see whether the user has read permission

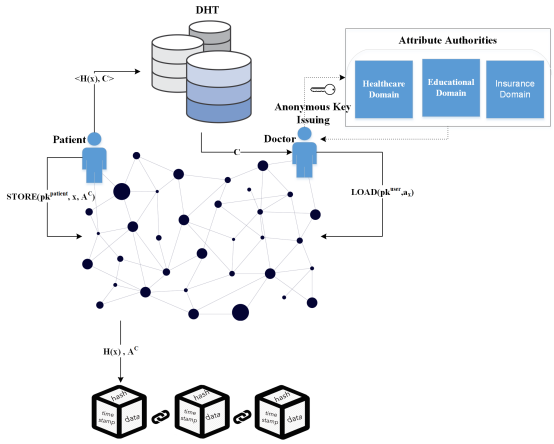

Figure 2. System Design

(Algorithm 4 line 4). If so, the process will become similar to that of loading PHR (Algorithm 4 line 9). Otherwise, a check is run to ensure that the user holds a role that allows him to run computation over the encrypted PHR (Algorithm 4 line 5). Once this is verified by a large enough number of endorsing peers, the transaction is stored in the ledger and the result of the computation is returned to the user. (See Algorithm 4 line 6.)

V. Evaluation of the Framework

A. Security Analysis

The security of our proposed framework depends on the security of the underlying components:

The security of the CA-less MA-ABE scheme is proven in [7]. More specifically, privacy is guaranteed as the data encrypted with this scheme remains confidential to unauthorized users. The scheme is also resistant to user collusion as well as attribute authority collusion of up to $N-2$ authorities

Algorithm 1 Checking Permission

procedure CHECKPERMISSION(pk_{sig}^{user}, $\mathbb{A}^C$, $\{d_k\}$)
  for all attribute authorities $k \in \{1..N\}$, **do**
    if $|\mathbb{A}_k^C \cap \mathbb{A}_k^{user}| \leqslant d_k$ **then**
      return $False$
    end if
  end for
  return $True$
end procedure

Algorithm 2 Storing PHR

procedure STORE($pk_{sig}^{patient}$, pk_{sig}^{writer}, x, $type_x$, $\mathbb{A}^C$,$\{d_k\}$)
  $a_x = \mathcal{H}(x)$
  $L[a_x] \leftarrow \mathbb{A}^C, \mathcal{H}(x), pk_{sig}^{patient}, pk_{sig}^{writer}, type_x$
  $C \leftarrow Encrypt(x)$
  $DHT[a_x] \leftarrow C$
  return a_x
end procedure

Algorithm 3 Loading PHR

procedure LOAD(pk_{sig}^{user}, a_x,$\{Sign(sk_{AA_k}, |\mathbb{A}_k^{user}|)\}$)
  $\mathbb{A}^C \leftarrow L[a_x]$
  $\{d_k\} \leftarrow L[a_x]$
  if CHECKPERMISSION($pk_{sig}^{user}, \mathbb{A}^C, \{d_k\}$) $= True$ **then**
    return $DHT[a_x]$ ▷ encrypted PHR is loaded iff user has the right set of attributes
  end if
end procedure

Algorithm 4 Secure Computation over PHR

procedure SECURECOMPUTE(pk_{sig}^{user}, a_x, $f(.)$, $\{Sign(sk_{AA_k}, |\mathbb{A}_k^{user}|)\}$)
  $\mathbb{A}^C \leftarrow L[a_x]$
  $\{d_k\} \leftarrow L[a_x]$
  if CHECKPERMISSION($pk_{sig}^{user}, \mathbb{A}^C, \{d_k\}$) $= False$ **then**
    if $Researcher \subset \mathbb{A}^{user}$ **then**
      return $f(DHT[a_x])$ ▷ computes over the encrypted PHR
    end if
  else
    LOAD(pk_{sig}^{user}, a_x,$\{Sign(sk_{AA_k}, |\mathbb{A}_k^{user}|)\}$)
  end if
end procedure

in each domain.

The blockchain component, being a tamper-proof distributed ledger, guarantees against manipulation of access policies and further protects data owners' privacy from adversaries.

B. Scalability Analysis

Typically, medical data management systems are governed by a jurisdictional body, and the number of those managed are in millions. As will become clear shortly, this is well within the scale that the architecture can handle.

We will discuss the scalability of the platform with respect to its main functionalities: *Key-issuing*, *Store*, and *Load*.

The complexity of the Key-issuing protocol is a function of the squared number of attribute authorities and the number of user attributes, i.e. $O(n^2|\mathbb{A}_{user}^k|)$, that are both small in practice. With that being said, the keys can be issued in an offline phase and therefore, can be disregarded in the scalability analysis.

The *Load* is a read-dominant transaction, as it accesses the ledger and DHT. Given that the ledger reads are served locally by a CouchDB database in Hyperledger Fabric, which is highly optimized for lookups, ledger read latencies are envisioned as negligible. This argument is consistent with the findings in [12] that suggest for read-dominant workloads, read throughput should scale linearly for the range of transaction rates that are substantially beyond that of our use-case. [1]

With relevance to the DHT reads, it should be pointed out that in high-performance DHT implementations such as Kademlia[13], the read operation takes only $\log n + c$ time, with n being the number of nodes in the blockchain network, and c being a small constant and hence, is scalable.

Ledger and DHT updates are the two primary operations in a *Store* transaction and thus, it can be regarded as a write-dominant transaction. In the experiments conducted in [12], the transaction latency has been reported to be less than 2 seconds for any write-dominant workload with less than 1000 tx/sec. It has also been shown that for write-dominant workloads, the throughput rose almost linearly until a load of 1000 tx/sec. These results, along with the fact that the arrival rate of *Store* transactions is roughly estimated at 7 tx/sec[2] suggests that *Store* transactions can be scaled.

Another interesting finding relevant to the previous point is that the write-dominant workload performed almost the same as the null workload, i.e. with no read or write transactions, up until a load of 1000 tx/sec. Since our estimated workload falls within this range, it could indicate that delays should be traced back to the three phases of the transaction before commit; namely endorsement, ordering, validation. These

[1] 450 transactions per second for the Ontario province, estimated using the stats reported in https://www.sciencedirect.com/science/article/pii/S1532046411000098#f0015

[2] estimated using the stats reported in https://link.springer.com/book/10.1007\%2F978-3-319-30146-4

phases can be configured in Hyperledger Fabric, to meet specific performance requirements.

VI. Conclusion and Future Work

In this work, we have proposed a novel access control mechanism in blockchain-based applications by incorporating attribute-based encryption scheme, through which we have not only achieved a strong privacy guarantee, but also extended the applicability of blockchain technology to much broader use cases.Although our solution is generic, we have used a medical data management system as the case study for this paper.

The use of blockchain has automated the data sharing process and has removed unnecessary intermediations. Therefore, with our blockchain-powered medical data management system, healthcare providers can gain access to PHRs without any disruptions from intermediaries. On the other hand, patients can share their healthcare data with whomever they want, with strong security guarantees.

To factor in privacy legislation, the usability of the platform should be extended to the cases where the raw data is not shared with certain roles. Instead, what is granted is the permission to run computations over data. Beyond any doubt, advanced learning algorithms can only perform well if they are provided with huge amounts of training data. With that said, strict privacy regulations concerning the sharing of personal health data seem to be a large roadblock to the future success of research in the medical domain, that could otherwise benefit from those algorithms. There is an extensive amount of literature on secure computation over private data, many of which even consider nonlinear calculations used in machine learning algorithms [14], [15]. However, our framework has a different setup in that, firstly, the data is assumed to be in an encrypted format. Furthermore, data files are stored in full and are not partitioned and distributed across various nodes. Even though homomorphic computation can perform computation over encrypted data, its applicability is very limited. Future work could be done to further study this problem.

References

[1] S. Nakamoto. (2008) Bitcoin: A peer-to-peer electronic cash system. Accessed on 10.04.2019. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *Proceedings of Security and Privacy Workshops (SPW), 2015 IEEE*. IEEE, 2015, pp. 180–184.

[3] U. U. Uchibeke, S. H. Kassani, K. A. Schneider, and R. Deters, "Blockchain access control ecosystem for big data security," *arXiv preprint arXiv:1810.04607*, 2018.

[4] A. Ekblaw, A. Azaria, J. D. Halamka, and A. Lippman, "A case study for blockchain in healthcare:"medrec" prototype for electronic health records and medical research data," in *Proceedings of IEEE open & big data conference*, vol. 13, 2016, p. 13.

[5] R. Xu, Y. Chen, E. Blasch, and G. Chen, "Blendcac: A blockchain-enabled decentralized capability-based access control for iots," in *Proceedings of 2018 IEEE International Conference on Blockchain*. IEEE, 2018, pp. 1027–1034.

[6] M. Li, S. Yu, K. Ren, and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in *Proceedings of International conference on security and privacy in communication systems*. Springer, 2010, pp. 89–106.

[7] M. Chase and S. S. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 121–130.

[8] A. Mense and M. Flatscher, "Security vulnerabilities in ethereum smart contracts," in *Proceedings of the 20th International Conference on Information Integration and Web-based Applications & Services*. ACM, 2018, pp. 375–380.

[9] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2005, pp. 457–473.

[10] M. Chase, "Multi-authority attribute based encryption," in *Proceedings of Theory of Cryptography Conference*. Springer, 2007, pp. 515–534.

[11] A. O. Yuta Namiki, huikang. (2018) hyperledger-fabricdocs documentation. Accessed on 10.04.2019. [Online]. Available: https://media.readthedocs.org/pdf/hyperledger-fabric/release-1.3/hyperledger-fabric.pdf

[12] A. Baliga, N. Solanki, S. Verekar, A. Pednekar, P. Kamat, and S. Chatterjee, "Performance characterization of hyperledger fabric," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 2018, pp. 65–74.

[13] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *International Workshop on Peer-to-Peer Systems*. Springer, 2002, pp. 53–65.

[14] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *Proceedings of 2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 19–38.

[15] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," *arXiv preprint arXiv:1506.03471*, 2015.