IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON HEALTHCARE INFORMATION TECHNOLOGY FOR THE EXTREME AND REMOTE ENVIRONMENTS

Received April 26, 2019, accepted May 14, 2019, date of publication May 17, 2019, date of current version June 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2917555

Blockchain for Secure EHRs Sharing of Mobile Cloud Based E-Health Systems

DINH C. NGUYEN[1], PUBUDU N. PATHIRANA[1], (Senior Member, IEEE), MING DING[2], (Senior Member, IEEE), AND ARUNA SENEVIRATNE[3], (Senior Member, IEEE)

[1]School of Engineering, Deakin University, Waurn Ponds, VIC 3216, Australia
[2]Data61, CSIRO, Kensington, WA 6152, Australia
[3]School of Electrical Engineering and Telecommunications, University of New South Wales (UNSW), Sydney, NSW 2052, Australia

Corresponding author: Dinh C. Nguyen (cdnguyen@deakin.edu.au)

This work was supported in part by the CSIRO Data61, Australia.

ABSTRACT Recent years have witnessed a paradigm shift in the storage of Electronic Health Records (EHRs) on mobile cloud environments, where mobile devices are integrated with cloud computing to facilitate medical data exchanges among patients and healthcare providers. This advanced model enables healthcare services with low operational cost, high flexibility, and EHRs availability. However, this new paradigm also raises concerns about data privacy and network security for e-health systems. How to reliably share EHRs among mobile users while guaranteeing high-security levels in the mobile cloud is a challenging issue. In this paper, we propose a novel EHRs sharing framework that combines blockchain and the decentralized interplanetary file system (IPFS) on a mobile cloud platform. Particularly, we design a trustworthy access control mechanism using smart contracts to achieve secure EHRs sharing among different patients and medical providers. We present a prototype implementation using Ethereum blockchain in a real data sharing scenario on a mobile app with Amazon cloud computing. The empirical results show that our proposal provides an effective solution for reliable data exchanges on mobile clouds while preserving sensitive health information against potential threats. The system evaluation and security analysis also demonstrate the performance improvements in lightweight access control design, minimum network latency with high security and data privacy levels, compared to the existing data sharing models.

INDEX TERMS Electronic health records (EHRs), EHRs sharing, mobile cloud computing (MCC), Internet of Medical Things (IoMT), blockchain, smart contracts, access control, privacy, security.

I. INTRODUCTION

Recently, there has been a growing interest in employing the blockchain technology to promote medical and e-health services [1]–[3]. Blockchain with its decentralized and trustworthy nature has demonstrated immense potentials in various e-health sectors such as secure sharing of Electronic Health Records (EHRs) and data access management among multiple medical entities [4]–[6]. Therefore, the adoption of blockchain can provide promising solutions to facilitate healthcare delivery and thus revolutionize the healthcare industry.

With the emergence of innovative technologies, including Mobile Cloud Computing (MCC) and Internet of Medical Things (IoMT), the healthcare industry has witnessed significant changes in e-health operations [7], [8]. Patients now can collect their personal health information at home based on mobile devices (such as smartphones and wearable sensors) and share on cloud environments where healthcare providers can access instantly to analyze medical records and provide timely medical supports. This smart e-health service allows healthcare providers remotely monitor patients and offer ambulatory care at home, which not only facilitates healthcare delivery but also brings economic benefits to patients. Further, the availability of complete EHRs on clouds also helps healthcare providers track patient health and offers proper medical services during diagnosis and treatment processes [9].

Besides all these great advantages, however, the trend of EHRs storage on clouds also poses security challenges which hinder the deployment of e-health applications on clouds [10], [11]. Among such security issues is secure

The associate editor coordinating the review of this manuscript and approving it for publication was Sabah Mohammed.

EHRs sharing between patients and healthcare providers on mobile cloud environments. Unauthorized entities may gain malicious access to EHRs without consent of patients, which has detrimental impacts on data integrity, privacy and security of cloud e-health systems [12]. Moreover, patients may find it difficult to track and manage their health records shared among healthcare providers on clouds. It therefore is necessary to propose efficient access control solutions for mobile cloud EHRs sharing systems.

Traditional access control approaches [14]–[16] for EHRs sharing assume that cloud servers are fully trusted by data owners and enable the servers to perform all access control and authentication rights on data usage. However, this assumption no longer holds in mobile clouds since the cloud server is honest but curious. The cloud sever will honestly perform the data requests, but meanwhile will obtain personal information without consent of users, which leads to serious information leakage issues and network security, accordingly. More importantly, conventional access control systems mainly rely on a predefined point of access, i.e. a centralized cloud server, and this can lead to the central point of failure for e-health networks [25].

Meanwhile, blockchain-based access control provides various new security features for e-health with great advantages over conventional access control solutions. First, the blockchain constructs immutable ledgers of transactions for data sharing system [5]. This means that transactions recorded in the blockchain cannot be modified or altered by any entities and transactions are only written to blockchain while recovery actions are not permitted. This guarantees high *system trustworthiness* and *integrity*. Second, access control using blockchain can achieve the *transparency* property with the ability of solving effectively the issue of data leakage which can be caused by curious servers. Any illegal access of servers and other entities to data storage will be reflected on the blockchain and broadcast to all network participants. In this way, any blockchain users can control data access and detect such malicious transactions for preventive actions. Third, the use of blockchain-based smart contract [21] can achieve the *authentication* and *user verification* property. By enforcing strict access control policies, smart contracts can authorize effectively user access to health data storage as well as detect and prevent effectively potential threats to health networks in a distributed manner. Final, blockchain coupled with the smart contract technology eliminates the reliance on central servers to ensure *fairness* among transaction parties. As the smart contracts are public on the blockchain, all the connected entities on the blockchain network will have a copy of them, which provide an equal right to control all contract operations. Specially, blockchain-based access control with its distributed nature can work well when any party fails without loss of data, risks and *trust* concerns [5].

Motivated by such advantages of blockchain, in this paper, we propose a new EHRs sharing model on a mobile cloud platform based on the blockchain technique. The foundation of our proposed system is a user access control framework to manage data access from network entities. Access control mechanisms are capable of effectively restricting illegal access to EHRs resources, while ensuring fast data retrieval for authorized entities.

A. THE PAPER MAKES THE FOLLOWING CONTRIBUTIONS

- We propose a novel EHRs sharing architecture based on blockchain and decentralized storage interplanetary file system (IPFS) for an e-health system on a mobile cloud platform. To improve security of EHRs sharing, we develop a trustworthy access control mechanism using smart contracts. Further, a data sharing protocol is designed to manage user access to our EHRs system.
- We investigate the performance of the proposed EHRs sharing model through usability tests on a mobile Android application and cloud computing provided by Amazon Web Services (AWS). The evaluation results demonstrate that the proposed method is feasible to various e-health scenarios.
- We provide a security analysis and extensive evaluation in various performance metrics to highlight the advantage of the proposed framework over current EHRs sharing solutions.

The remainder of our paper is organized as follows. The Section II introduces state-of-the-art studies towards the use of access control solutions and blockchain for EHRs sharing on clouds. The Section III introduces blockchain concepts and its main components that will be utilized in this paper. In the Section IV, we propose a system model for the EHRs sharing scheme using blockchain and smart contracts, and design objectives are also presented. Next, we present a prototype implementation of decentralized access control for EHRs sharing in a mobile cloud blockchain network in the Section V. Smart contract design and data sharing protocol are also analyzed. The experimental results are discussed in the Section VI, while security analysis and system evaluations are given in the Section VII. Finally, our conclusions are drawn in Section VIII.

II. RELATED WORK

There have been several traditional solutions to deal with the problem of secure EHRs sharing on cloud environments. The work [14] proposed a broker based access control scheme for selective EHRs sharing among various healthcare providers in cloud computing. Besides, they only investigated their sharing concept on a computer simulation with Virtual Machines (VMs) while neglecting implementation on resource-constrained devices like smartphones. To maintain authentication for medical users and the EHRs sharing system, Public Key Infrastructure (PKI) was employed on cloud to ensure security standards of e-health systems [15]. Further, the authors in [16] employed an attribute authority for granting keys for data consumers in the attribute-based encryption (CP-ABE) prototype to achieve fine-grained access control for EHRs sharing on cloud. They evaluated the

proposed scheme via a numerical simulation by deploying the experiment environment on the Ubuntu Linux Desktop, and the capability of decentralized access was ignored in this research.

In the context of blockchain technology, various studies have investigated the capability of blockchain to support e-health data sharing. Blockchain was exploited in [17] to ensure reliable EHRs accessibility for medical users. The authors relied on smart contracts to manage EHRs usage of doctors. The authors focus on theoretical analysis and therefore, the feasibility of the proposed solution had not been confirmed in real EHRs sharing scenarios. As a result, some important features of EHRs sharing such as flexibility, availability and identity management had not been investigated. Meanwhile, to ensure secure EHRs exchange among medial consumers, a data management concept based on blockchain was introduced in [18]. Blockchain enabled decentralized strategy was also proposed to solve storage issues of large medical records. The work [19] introduced a system called MeDShare which dealt with the problem of medical data sharing among cloud service providers. Smart contracts were employed to design an access control mechanism that was capable of tracking data exchanges between untrusted parties with the ability to detect malicious access to EHRs storage, while providing provenance and auditing on medical data. The performance of the proposed system was evaluated only through network latency measurements with theoretical analysis, showing that the proposal can be a solution to achieve efficient data sharing without risks on data privacy. The authors in [20] introduced an innovative user-centric health data sharing method that could enhance identity management and preserve data privacy for patients. They had not considered the problem of identity management and user authentication to protect data storage.

To better preserve data privacy and availability on clouds, the authors in [21] designed a decentralized storage system by combining Interplanetary File System (IPFS), Ethereum blockchain, and attribute-based encryption (ABE) technologies. In their model, a data owner distributes secret key for users, and encrypts his data under predefined access policy to achieve fine-grained access control on cloud data. The smart contracts were designed to implement the keyword search in the decentralized storage systems that could address the problem of not returning search results honestly in traditional cloud storage. The authors evaluated the system scheme on the Ethereum official test network Rinkeby, and the performance was examined through smart contract cost tests and security analysis were also provided to show an improvement in data management, data search fairness on cloud and data privacy. To solve the problem of storing large data on blockchain, the works in [22]–[24] introduced models that integrated the IPFS system with smart contracts to provide data sharing in IoT scenarios. These architectures facilitate IoT communication and data exchanges in untrusted environments.

Despite the promising results, these existing works still show several limitations. First, the designs of conventional access control strategies [14]–[16] require external key authorities with a Public Key Infrastructure (PKI), which can be extremely complex and usually demands expensive resources to achieve secure EHRs sharing [25]. Besides, these centralized EHRs systems can cause excessive delay in offering medical services when multiple entities request medical information. Second, with these existing approaches, patients have little control over their personal data on cloud EHRs systems. Current EHRs sharing schemes can pose challenges for patients in keeping track of the exchange of their health records and specifying who should have the access to their medical records. In practical scenarios, some authorized healthcare practitioners may try to use health data of patients for their illegal purposes, which can lead to leakage of the sensitive patient information. Final, the capability of access control on EHRs sharing in blockchain has not been investigated in real scenarios.

Unlike pioneering studies [17]–[20], this paper not only considers access control over data uploading and sharing, but also provides an effective data management strategy on mobile cloud with smart contract and blockchain. It is worth noting that the feasibility of blockchain based access control was demonstrated in various practical scenarios, such as IoT [26]–[29], vehicular edge computing and networks (VECONs) [30]. However, we find that a study on a comprehensive EHRs sharing solution with data management on mobile cloud is still missing. This work aims to fill the gap that considers EHRs sharing issues and access control on data usage in e-health blockchain. The main differences between our study and the existing EHRs sharing schemes [17]–[20] can be highlighted as follows.

1) We build a comprehensive data offloading and data sharing architecture using blockchain for mobile cloud e-health applications. To protect health database, we propose a data storage system by leveraging IPFS technology. Specially, we implement a smart contract design on an Ethereum blockchain platform on Amazon cloud, aiming to exploit the access control capability of smart contract and blockchain to manage authentication, user identification and ensure system integrity.
2) Instead of theoretical analysis as current studies, in this work, we focus on implementing a real data sharing design using mobile devices and mobile clouds on blockchain. Based on implementation results, we identify many useful technical features of blockchain for EHRs sharing, which can make important contributions to blockchain research in IoT applications including healthcare.
3) We provide a comprehensive evaluation of the proposed framework in terms of various technical aspects including access control performance, network latency, and security. Challenges and open issues of blockchain implementation for EHRs sharing and e-health systems are also provided.

III. PRELIMINARIES

In this paper, we employ an Ethereum blockchain platform for building our e-health system [31], [32]. Typically, Ethereum is a new distributed blockchain network similar to most platforms such as Bitcoin [33]. A big advantage of Ethereum is its adaptable and flexible features, which allow to build any blockchain applications such as e-healthcare. We review main components of an Ethereum network which will be used in our design.

Like Bitcoin, the Ethereum platform has a blockchain, which contains blocks of data transactions and smart contracts. Blocks are verified and appended to the blockchain by an algorithm called proof of work by miners to achieve a secure, tamper-resistant consensus among all nodes in the network. Each block contains hash of its previous block in a chronological order. The cryptography hash algorithm on the platform guarantees that linked blocks are resistant to be modified and blockchain data is immutable.

1) ETHEREUM ACCOUNT

Ethereum has two different accounts: externally owned accounts (EOAs) and contract accounts. Every account is indexed by a 20-byte address and defined by a pair of keys, a private key and a public key. To interact with Ethereum blockchain, each user needs to create an account to become an entity in the network.

2) SMART CONTRACT

A smart contract [32] is a kind of self-operating computer program, which can be executed automatically when specific conditions are met. In the Ethereum blockchain, a smart contract is a special account, which contains data and code with multiple programmable functions. Users can use their Ethereum account to interact with smart contracts via application binary interfaces (ABI). Functions defined in smart contracts can be triggered by a new transaction sent from an account. This property allows entities to implement their job functionalities such as data transmission, request handling or access management.

3) TRANSACTION

An Ethereum transaction is a data packet to transfer ether (Ethereum native token) from an account to another. A typical structure of transaction is as follow *{Account Nonce, address of recipient, Gas price, Gas limit, Ether value, sender's signature, data field}*. Below is an example of a transaction. *Transaction Tx = new Transaction(getFromAddress(), gasPrice, gasLimit, to, value, data)*. In our model, we use the *data field* to declare request IDs to smart contracts for data request on cloud storage.

IV. SYSTEM MODEL

In this section, we present a system architecture and introduce the concept of data uploading and data sharing in our system. Further, design goals in this paper are also highlighted.

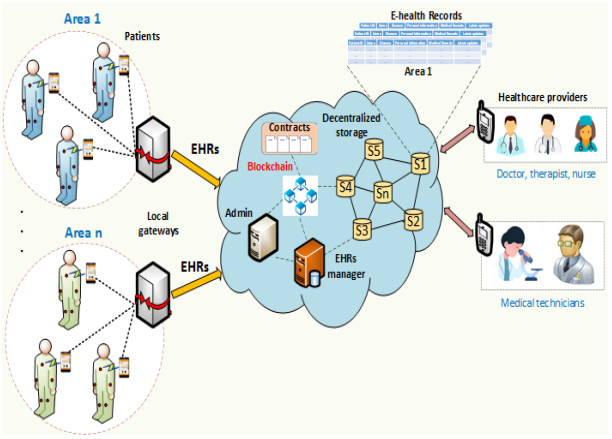

FIGURE 1. The overview of blockchain based e-health system on mobile cloud.

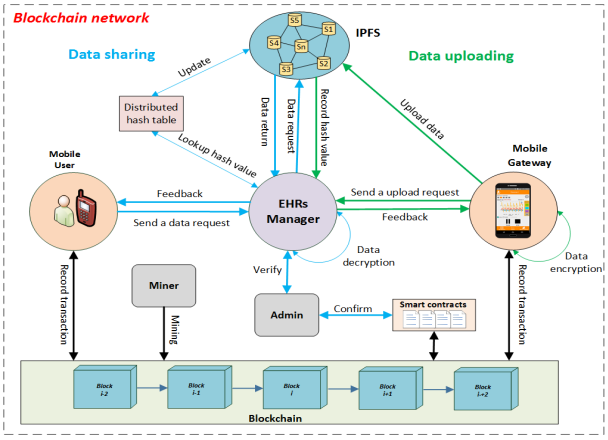

FIGURE 2. The data flow of the proposed mobile cloud blockchain system.

A. SYSTEM ARCHITECTURE

We consider an e-health scenario on a mobile cloud platform where patient records are gathered from a network of local gateways and stored on a public cloud for sharing with healthcare providers as shown in Fig. 1. E-health records may include personal information and medical history which are provided by patients. Patients have their own patient ID P_{ID} and are classified based on their current living area with an area ID A_{ID}. In this model, we assume that the wearable sensor network is private and managed by its local user (patient). We also assume that EHRs can be collected from wearable body sensors by a mobile application integrated in patients' smartphone. Therefore, the address of a patient on blockchain can be formulated as $Addr = \{A_{ID}, P_{ID}\}$. Because it is infeasible to store medical data on blockchain, we suggest to only keep addresses of patients on blockchain, while large medical records are stored on decentralized cloud storage. Further, to manage medical records, a cloud EHRs manager *ME* is proposed. Thus, in order to retrieve a certain health record on cloud, a participating entity needs to know patient addresses which are visible on the blockchain network. The data flow of the proposed mobile cloud blockchain system is also shown in Fig. 2.

Next, we assume that each health provider (such as a doctor, physician or a nurse) has a mobile phone to retrieve EHRs on the cloud with their ID HP_{ID}. By this way, they can acquire medical history on cloud storage for their medical analysis. For example, a doctor can access EHRs of a patient to analyze medical records and diagnose health problems for providing proper healthcare services.

We also develop a cloud blockchain network for EHRs sharing. An Ethereum platform is chosen thanks to its advantages mentioned in the previous section. The main components of cloud blockchain network are presented as follows.

- **EHRs manager:** The EHRs manager plays a significant role in our data sharing framework. It is responsible to control all user transactions on the blockchain network, including data storage processes of mobile gateways and data access of mobile users. The management capability of EHRs manager is enabled by smart contracts through strict user policies.

- **Admin:** It is used to manage transactions and operations on cloud by the means of adding, changing or revoking access permissions. Admin is responsible to deploy smart contracts and the only entity with the ability to update or modify policies in smart contracts.

- **Smart contracts:** They define all operations allowed in the access control system. Users can interact with smart contracts by the contract address and Application Binary Interface (ABI). Smart contracts can identify, validate request and grant access permissions for medical users by triggering transactions or messages. The smart contract and its operations are accessible to all blockchain entities. It is considered as core software in our e-health platform.

- **Decentralized storage:** Since it is infeasible to share and store large data on blockchain, we leverage a decentralized peer-to-peer file system InterPlanetary File System (IPFS), a very promising solution to build a file sharing platform in the blockchain network [34]. IPFS has been built on top of the BitTorrent protocol and the Kademlia Distributed Hash Table (DHT). In the IPFS system, the role of central server is eliminated, and users can store data in a network of distributed storage nodes on the same file system with advantages over existing cloud storage such as no single point of failure, high storage throughput and data retrieval improvement [35]. By using IPFS, users can identify and access files by relying on the cryptographic hashes of their contents.

In the EHRs sharing context in our paper, medical records are encrypted and stored in IPFS nodes, while their hash values are recorded by EHRs manager and stored in DTH. We also integrates smart contracts with IPFS to improve decentralized cloud storage and controlled data sharing for better user access management. We configure the EHRs storage running on the IPFS platform with a network of individual nodes $S = \{S_1, S_2, \ldots, S_n\}$, each node only stores EHRs of patients in a certain area as shown in Fig. 3.

- **Data block structure:** The Fig. 4 shows a structure of EHRs block with the following main components.

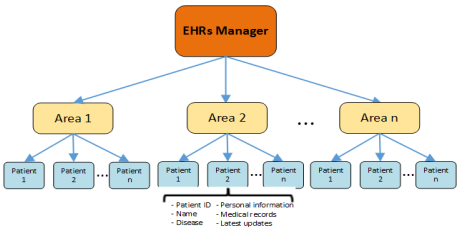

FIGURE 3. Structure of cloud EHRs storage system.

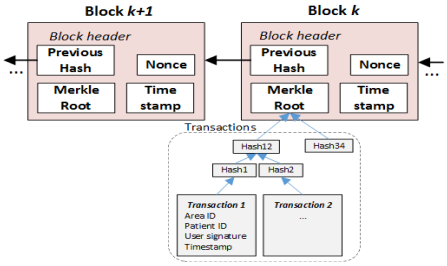

FIGURE 4. The data block structure.

- *Transaction records:* Transactions in our block are organized in a Merkle tree based structure where a leaf node represents a data access transaction of mobile users. To provide a data request, a mobile user needs to provide patient information (*Area ID, Patient ID*) to establish a transaction that is also signed with user's private key at a certain time (timestamp). This digital signature aims to establish trust between the user and cloud server.

- *Block header:* The block header contains the following metadata to verify the data block.

- **Hash:** The SHA256 hash of the block. With an example in Fig. 4, the hash value can be formulated as $Hash12 = Hash(Hash1 + Hash2) = Hash[(Tx1.Hash) + (Tx2.Hash)]$.
- **Previous hash:** The hash of the previous block that is used for block validation.
- **Merkle Root:** A structure to store a group of transactions in each block.
- **Nonce:** It refers to a number that is generated by proof of work operation on miner nodes, in order to produce a hash value below a target difficulty level.
- **Timestamp:** It refers to the time of when the block was created. It is also the timestamp of the last transaction in the block.

B. THE DATA UPLOADING AND SHARING FRAMEWORK ON MOBILE BLOCKCHAIN

In this subsection, we introduce two mechanisms for secure data uploading and data sharing by leveraging smart contracts and blockchain. The use of smart contract allows to manage automatically transactions and operations on blockchain with high efficiency and reliability. Specially, at each mobile device, we design a blockchain client module, which implements a full functionality to participate in the

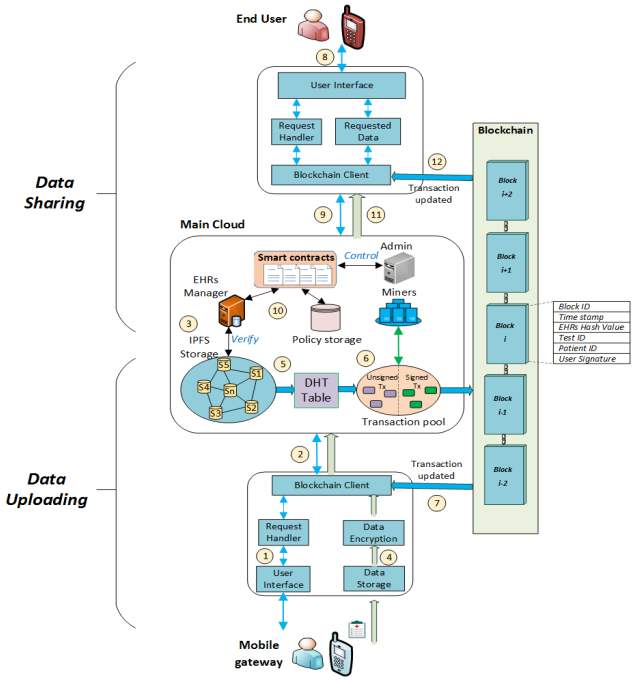

FIGURE 5. The data uploading and data sharing processes on mobile cloud blockchain.

Area ID	Patient ID	Hash value of data file
A101	P1001	Qm690ebb015f1c9d9...
	P1002	Qm40b3643d2facf894...
	P1003	Qm435039fc72817c61...
	...	...
A201	P2001	Qmd6f71b01b9f987b5...
	P2002	Qm06bcab5ff755d822...
	P3003	Qm861c8f5ec34e4ba3...
	...	...
...	...	...

FIGURE 6. The structure of distributed hash table (DHT) in our IPFS storage design.

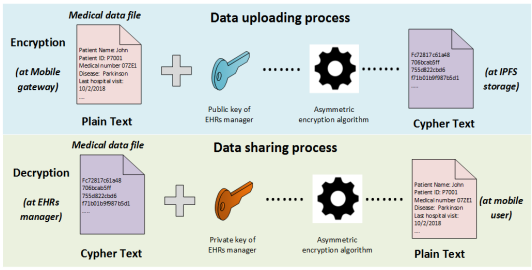

FIGURE 7. Data file encryption and decryption in our design.

cloud blockchain network. This module is responsible to encode transactions and data requests, sign digitally on transactions and connect with blockchain for transaction tracking. Besides, the user interface module is designed for user interaction, and a request handler module for processing user request information. The work flow of EHRs uploading and sharing processes is illustrated in Fig. 5. The description of each step number is summarized as follows.

① A mobile gateway initializes a request as a new transaction for uploading data to cloud server.
② The blockchain client processes and sends the request to EHRs manager and smart contracts for verification.
③ The EHRs manager verifies the request by smart contracts with a strict control policy. If the request is accepted, a response will be returned to the gateway for uploading data.
④ The gateway now can select EHRs, i.e. a health data file collected from wearable sensors and encrypt data with the public key of EHRs manager. Then the gateway uploads this encrypted file to IPFS storage on cloud.
⑤ The IPFS storage will store the data file into a corresponding storage node using uploading information (Area ID, Patient ID) and automatically returns a hash value which is kept in the DHT table (see Fig. 6).
⑥ Uploading transactions are grouped into data blocks, which are then inserted into the transaction pool for confirmation by miners to append to the blockchain.
⑦ The uploading transaction is updated at the mobile gateway for tracking.
⑧ The mobile user joins the blockchain network, prepares a transaction with request information (*requestIDs*), and signs it with the private key *SK* for data access on cloud.
⑨ The user connects with cloud and sends a data access request to cloud using the blockchain client module.
⑩ The EHRs manager will verify the access right of user via the access control strategy with smart contract. Once the access request is confirmed, the EHRs manager will analyze the *requestIDs* and forwards such information to IPFS storage for retrieving data.
⑪ The EHRs manager decrypts such requested data using an asymmetric encryption algorithm (see Fig. 7) and returns this requested data to the requestor.
⑫ The uploading transaction is updated at the mobile device of user for tracking via the blockchain client.

The concept of data uploading and data sharing will be explained in the next subsection in details.

1) DATA UPLOADING PROCESS

The EHRs uploading strategy is shown in the following stages:

Stage 1 Initialization (Executed by the Mobile Gateway): The mobile gateway such as a smartphone needs to create a blockchain account to join the blockchain network and initialize a request as a transaction for uploading data. The mobile gateway uses the blockchain client module to interact with blockchain on clouds. In the data uploading, this module will create a storage transaction *Ts* as a request with information index including request metadata (Gateway ID), digital signature and timestamp for verification as well as patient index for storage. The request then will be sent to the EHRs manager on cloud for verification via wireless network. (Steps 1, 2 in Fig. 5).

Stage 2 Verification and Uploading EHRs Data to IPFS Storage (Executed by the EHRs Manager):

After receiving a transaction from gateway, the EHRs manager will issue a signal to smart contract as a notification of a new data uploading request. By using the policy list in the policy storage, the smart contract can verify the transaction and if it is accepted, a message will be returned to gateway via the blockchain client to allow to upload data (Step 3 in Fig. 5).

Once the request is approved, the mobile gateway now can encrypt the EHRs file with the public key of EHRs manager using an asymmetric encryption algorithm (see Fig. 7). The encrypted file is then uploaded to the IPFS cloud storage server which is connected with the blockchain client module of gateway (Step 4 in Fig. 5). The IPFS storage server will automatically return a hash of this uploaded file to the EHRs manager and this hash value is also updated in DHT table. Note that in this context, this hash is produced by file content as the IPFS storage concept [34], and any modifications on data files in IPFS can be easily detected by the EHRs manager. The encrypted file is now available on the IPFS network. Each IPFS node stores data files of patients in a certain area as configured in the system model (see Fig. 1).

Stage 3 Adding the Storage Transaction to Blockchain (Executed by Miners):

Besides adding EHRs to cloud storage, metadata of uploading transactions (Area ID, Patient ID and hash value as shown in Fig. 5) is also inserted into the unsigned transaction pool. The miners will form periodically transactions in the pool into blocks for mining. The fastest miner which verifies the data block will send the signature to other miners for validation. If all miners achieve an agreement, the validated block with its signature is then appended to the blockchain in a chronological order. Finally, all network users receive this block and synchronize the copy of the blockchain via the blockchain client. The work flow for the above process is shown in Steps 6, 7 in Fig. 5. With our blockchain design, any entities can monitor transactions in a distributed manner. More importantly, due to storing hash values instead of raw data on blockchain, the proposed model avoids risks of data leakage and thus ensures high data privacy.

2) DATA SHARING PROCESS

The data sharing scheme with access control is designed to allow mobile users to access data on the main cloud. The process is explained in the following stages:

Stage 1 Generation of Request Information (Executed by Mobile User):

To access data on cloud storage, a mobile user needs to establish a user transaction *Tx* as a data request for accessing data. By creating a blockchain account, which includes a private key to sign the transaction and public key for user identification, users can access our blockchain. To request data, the user also needs to provide address information of patient as metadata for the request. The transaction *Tx* will then sent to the EHRs manager on cloud for access validation. These work flows are shown in Steps 8, 9 in Fig. 5.

Stage 2 Access Verification and Data Retrieval (Executed by the EHRs Manager):

The user transaction is sent to the EHRs manager for audit and verification. Then, the EHRs manager issues a message to the smart contract to verify for this request. Based on the policy list which includes user public keys, the smart contract will confirm and return a message to the EHRs manager. If the user public key is available in the policy list, the access permission is granted and now the requester can acquire data of their interest. Otherwise, the smart contract will issue a penalty to the requester and deny any further request of this user (Step 10 in Fig. 5).

Once the request right is approved, the EHRs manager will locate using request index. Note that request index includes storage address of patient data so that the EHRs manager can specify which storage node contains request data. The EHRs manager analyses the *requestIDs* (Area ID and Patient ID) of authorized access and looks up the hash value of data file containing request data in the distributed hash table to retrieve the requested data on IPFS storage. The IPFS system then returns the data file encrypted by the data uploading process. The EHRs manager uses an asymmetric encryption algorithm (see Fig. 7) with its private key to decrypt the encrypted data file, and returns the decrypted data to the requestor (Step 11 in Fig. 5).

Stage 3 Adding the Data Sharing Transaction to Blockchain (Executed by Miners):

Similar to the data uploading process, the transactions will be grouped into blocks and added to the transaction pool for mining by a network of miners. All verified transaction will be recorded and appended to blockchain for sharing with all mobile users. The end user can update the information of their transactions via the blockchain client that is integrated with their mobile phone (Step 12 in Fig. 5).

In the next section, we present a real prototype implementation on a mobile cloud blockchain application for the data sharing scheme to demonstrate the feasibility of our proposed scheme.

C. DESIGN GOALS

To enable the efficiency and security of EHRs sharing on mobile clouds under the proposed system architecture, our developed framework should achieve the following design objectives.

1) The system should have the properties of user identity and authentication. The framework ensures to strictly verify access from mobile to guarantee system trustworthiness. It should only allow authorized entities to access EHRs, while preventing effectively potential threats to the EHRs system.
2) The system should provide a fast data retrieval capability with an authorization design based on smart contracts. The user access control design should be lightweight to avoid high network latency. In fact, how to design a

highly trusted scheme with minimum system overheads is a challenging issue in designing any EHRs sharing systems.

3) The system should achieve high security levels, system integrity and data privacy and provide flexibility to mobile users, which can make the proposed system feasible to many healthcare scenarios.

To achieve these performance goals, we propose a trustworthy access control scheme with a data sharing protocol using a smart contract to manage user access to EHRs. A peer-to-peer cloud storage system based on IPFS is also designed to achieve decentralized data storage and controlled data access for EHRs sharing. These approaches can overcome challenges of centralized models with high access control performance and low system latency. Besides, we also design a mobile Android application for interactions of users with the cloud blockchain. Security analysis and design features are also analyzed to demonstrate the applicability of the proposed EHRs sharing system.

V. IMPLEMENTATION

We present a proof of concept (PoC) implementation of a decentralized access control for EHRs sharing on mobile clouds to investigate and evaluate the proposed system. The following subsections show details of our implementation with system configurations.

A. SYSTEM SETTING

We considered an EHRs sharing framework on mobile cloud as shown in Fig. 8. We deployed a private Ethereum blockchain network on the Amazon cloud computing [36] where two virtual machines AWS EC2 were employed as the miners, two virtual machines Ubuntu 16.04 LTS were used as the admin and EHRs manager, respectively. Our smart contract was written by Solidity programming language [37] and deployed on AWS Lambda functions. Each function interacts with the cloud blockchain via the web3.js API. Users can interact with smart contracts through their Android phone where a Geth client [38] (a command line interface implemented in the Go language) was installed to transform each smartphone into an Ethereum node. By using the Geth client, a mobile user can create an Ethereum account to communicate with our blockchain network for accessing data. The web3.js library [39], a lightweight Java library for working with smart contracts and blockchain, was also used for developing the mobile application to connect with the Ethereum blockchain network. In our experiment in this paper, we used two Sony mobile phones running on an Android OS version 8.0 platform to investigate results of EHRs sharing.

Next, we set up IPFS on Amazon cloud to build a decentralized storage system. Currently, IPFS can be built on top of the cloud computing on AWS [40]. Configurations for the IPFS system on AWS are shown in Fig. 9. In this paper, we build the medical records system by our mobile healthcare platform called *BioKin* [41] which includes a network of wearable sensors and a mobile Android application to collect

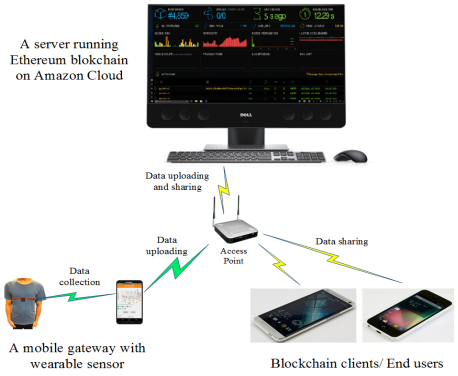

FIGURE 8. Experiment setup for EHRs sharing simulation.

```
AWS::CloudFormation::Init:
    config:
      files:
        '/tmp/IPFS_install.sh':
          content: |
            #!/bin/bash -e
            sudo yum install wget -y;
            wget https://dist.ipfs.io/go-ipfs/v0.4.17/go-ipfs_v0.4.17_linux-amd64.tar.gz;
            tar -xzvf go-ipfs_v0.4.17_linux-amd64.tar.gz;
            cd /go-ipfs
            sudo ./install.sh;
            ipfs init;
            ipfs config Addresses.API /ip4/0.0.0.0/tcp/5001
            ipfs config --json API.HTTPHeaders.Access-Control-Allow-Origin '["*"]'
            ipfs config --json API.HTTPHeaders.Access-Control-Allow-Methods '["GET",
"POST"]'
            ipfs config --json API.HTTPHeaders.Access-Control-Allow-Headers
'["Authorization"]'
            ipfs config --json API.HTTPHeaders.Access-Control-Expose-Headers '["Location"]'
            ipfs config --json API.HTTPHeaders.Access-Control-Allow-Credentials '["true"]'
            ipfs daemon &
            sleep 10
            ipfs bootstrap rm --all
            exit
          mode: '000731'
          owner: root
          group: root
      commands:
        install_ipfs:
          command: /tmp/IPFS_install.sh
```

FIGURE 9. Configuration for integrating IPFS on AWS cloud.

and process patient data for cloud storage. A Java library for uploading data to IPFS on cloud was also integrated with our mobile app [42]. In our test, it is assumed that medical records were collected by *BioKin* sensors and stored in the IPFS system. To encrypt medical data files in the mobile gateway, we employed an asymmetric encryption algorithm RSA that was written in Java and deployed on Android app [43]. Note that all data files were encrypted by the public key of EHRs manager in data uploading and then decrypted by EHRs manager with its private key in the data sharing process in an asymmetric manner as explained in Fig. 7.

B. ACCESS CONTROL FOR EHRS SHARING

In this subsection, we design a smart contract to formulate our access control model. We also design an access protocol that presents the work flow of EHRs sharing scheme.

1) SMART CONTRACT DESIGN

We first create a *dataSharing contract* controlled by the admin to monitor transaction operations in our blockchain network. We denote *PK* as the public key of user, *userRole* as the role of user, *Addr* as the address of patient in blockchain. The contract mainly provides the following five functions.

- *AddUser(PK, userRole):* (executed by Admin) This function allows to add a new user to the main contract. User is identified by their public key and is added into the contract with a corresponding role based on their request. User information is also kept in cloud storage as part of system database.

```
pragma solidity ^0.4.22;

contract DataSharing {
    bool public status;
    address public Admin
    address public EHRsManager;
    uint public numberOfUsers;
    mapping (address => uint) public userId;
    User[] public users;

    event UserAdded(address UserPK, string UserRole);
    event UserRemoved(address UserPK);
    event DataResult(address userPK);
    event ReturnAccessResult(
    address indexed _from,
    string _message,
    bool _result,
    uint _time,
    uint _penalty );

    struct User {
        address userPK;
        string userRole;
    }

    struct EHRstorage {
        string storageName;
        address AreaID;
        address PatientID;
        string Hash_value;
    }

    struct PolicyItem{ //for <EHRs cloud resource,
action>;
        address userPK;
        address AreaID;
        address PatientID;
        string permission; //verify permission: "allow" or
"deny"
        bool result; //return access result
        bool checkID; //check the access
    }
    mapping (bytes32=> EHRstorage) public DTHtable;
    mapping (bytes32 => mapping(bytes32 =>
PolicyItem)) policies; //////mapping (EHRs resource,
action) to policy checking

    modifier onlyAdmin {
        require(msg.sender == Admin);
    }
    function SC_creation (string EHRsContract) public {
        Admin = msg.sender;
        status = true;
        addUser(0, "", "");
        addUser(Admin, 'Creator of Smart Contract', "");
        numberOfUsers = 0;
    }
    function addUser(address userPK, string userRole)
onlyAdmin public {
        require(status = true);
        uint id = userId[userPK];
        if (id == 0) {
            userId[userPK] = users.length;
            id = users.length++;
        }
        users[id] = User({user: userPK, role: userRole});
        UserAdded(userPK, userRole, userNotes);
        numberOfUsers++;
    }

    function removeUser(address userPK)
onlyAdmin public {
        require(userId[userPK] != 0);
        for (uint i = userId[userPK]; i<users.length-1;
i++){
            users[i] = users[i+1];
        }
        delete users[users.length-1];
        users.length--;
        UserRemoved(userPK);
        numberOfUsers--;
    }
    function retrieveEHRs(string retrieve_name,
address userPK, string _storageName, address
_AreaID, address _PatientID, string _Hash_value)
onlyAdmin public {
        bytes32 key = stringToBytes32(userPK);
        /* Look up the EHRs data in DTH table */
        DTHtable[key].storageName = _storageName;
        DTHtable[key].AreaID = _AreaID;
        DTHtable[key].PatientID = _PatientID;
        DTHtable[key].Hash_value = _Hash_value;
        DataResult(key);
        return (string(DTHtable[key]))
    }

    function policyList(string _resource, string
_action, string _permission, address userPK,
address AreaID, address PatientID ) public{
        bytes32 EHRsresource =
stringToBytes32(_resource);
        bytes32 action = stringToBytes32(_action);
        if(msg.sender == EHRsManager){
        policies[EHRsresource][action].userPK = true;
        policies[EHRsresource][action].AreaID = true;
        policies[EHRsresource][action].PatientID =
true;
        policies[EHRsresource][action].permission =
_permission;
        policies[EHRsresource][action].result = false;
    }
    else throw;
    }
    function penalty(address userPK, string
_resource, string _action, uint _time) public{
        bool policycheck = false;
        bool behaviorcheck = true;
        bool checkID = 0;
        uint penalty = 0;
        if(msg.sender==EHRsManager){
        userPKcheck =
policies[EHRsresource][action].userPK;
        AreaIDcheck =
policies[EHRsresource][action].AreaID;
        PatientIDcheck =
policies[EHRsresource][action].PatientID;
        checkID = userPKcheck && AreaIDcheck &&
PatientIDcheck;
        if (checkID is true) // Detect an authorized
access
        ReturnAccessResult(msg.sender, "Successfull",
true, _time, penalty);
        else // Detect an unauthorized access
        ReturnAccessResult(msg.sender, "Failed!",
true, _time, penalty);
        }
    }
```

FIGURE 10. Code script of smart contract for EHRs sharing.

- *DeleteUser(PK, userRole):* (executed by Admin) It is used to remove users from the network based on the corresponding public key. All personal information is also deleted from cloud storage.

- *PolicyList(PK):* (executed by Admin) A peer of health provider-patient can agree on a policy which expresses their relation in medical services. For example, a patient has a unique doctor for his health care and only this doctor has rights to access EHRs of his patient. The policy list contains public key of all entities for identification when the smart contract processes new transactions.

- *RetrieveEHRs(PK, Addr):* (executed by EHRs manager) It allows to retrieve medical records on cloud storage. A network participant needs to provide the address of patient (including *Patient ID* and *Area ID*) to the smart contract. The contract then verifies and sends a message to EHRs manager to extract and return data to the requester.

- *Penalty(PK, action):* (executed by Admin) When detecting an unauthorized request to EHRs system, the EHRs manager will inform to smart contract to issue a penalty to the requester. In our paper, we give a warning message as a penalty to the unauthorized mobile entity.

The smart contract design on Ethereum blockchain can be seen in Fig. 10.

2) DATA SHARING PROTOCOL

To operate the user access control system for EHRs sharing, we also develop an access protocol which is performed when a mobile user executes a transaction to request EHRs on cloud. The access protocol is operated through four steps.

Step 1 Transaction Pre-Processing (Executed by the EHRs Manager):

The EHRs manager receives a new transaction *Tx* from a mobile user (such as a health provider or a patient). The EHRs manager will obtain the public key of the requester by using the *Tx.getSenderPublicKey()* function and send it to the contract for validation.

Step 2 Verification (Executed by the Admin):

After receiving a transaction with a user PK from EHRs manager *(msg.sender = ME)*, the admin will verify access rights of the requester based on its *PK* in the policy list of the smart contract. If the *PK* is available in the list, the request is accepted and now a data access permission is granted to the requester. Otherwise, the smart contract will issue a penalty to this request through the *penalty()* function. In this case, all EHRs access activities are denied and the request is discarded from the blockchain network.

Step 3 EHRs Retrieval (Executed by the Admin):

Once the access permission for the new transaction is granted, the contract will decode the transaction using the *abiDecoder:decodeMethod(Tx)* function [44] to obtain the address information of EHRs in the data field of transaction (see Section II). Now the admin can know the *Area ID* and *Patient ID* of EHRs for the transaction. The admin then sends a request to the EHRs manager for locating and extracting medical records in the decentralized cloud storage system.

Step 4 EHRs Feedback (Executed by the EHRs Manager):

Once the requested EHRs are found, the EHRs manager will send them to the requester via an off-chain network created for cloud-mobile user communication. Now the data process is finished and a new transaction is appended to blockchain and broadcast to all network entities. Note that data in such transactions are mainly patient addresses, which are lightweight and efficient to store on the blockchain. Therefore, our blockchain solution can be applied to many e-health applications.

The access protocol is summarized in Algorithm 1.

C. TESTING METHODOLOGY

Based on hardware and software settings, in this section, we concentrate on evaluating the proposed EHRs sharing system via a real test. We first deploy an Ethereum blockchain platform on Amazon cloud and connect with a blockchain client application which is integrated with Android mobile phones. For simplicity, in the cloud EHRs storage system, we only consider a testing area ($AreaID = 1$) with 10 patients ($PatientID = 1, 2, \ldots, 10$). That means in the IPFS storage system, we set up one storage node with 10 data files of different patients. Note that medical data files used in our test were collected by wearable sensors as shown in Fig. 8.

Algorithm 1 EHRs Access Protocol

Input: Tx
Output: *Result*
Initialization: *(by the EHRs Manager)*
1: Receive a new transaction Tx from a mobile user
2: Get the public key of the requester
$PK \leftarrow Tx.getSenderPublicKey()$
3: Send the public key to Admin
$(msg.sender = EHRsManager)$
Verification *(by the smart contract and Admin)*
4: **if** PK is available in the policy list
$policyList(PK) \leftarrow true$
5: **end**
6: Decode the transaction
$decodedTx \leftarrow abiDecoder.decodeMethod(Tx)$
7: Specify request information
$Addr \leftarrow web3.eth.getData(decodedTx([DataIndex])$
8: Specify $AreaID$ and $PatientID$
$A_{ID} \leftarrow Addr(Index[A_{ID}])$; $P_{ID} \leftarrow Addr(Index[P_{ID}])$
EHRs Retrieval *(by the smart contract)*
9: **while** true **do**
10: **if** $policyList(A_{ID}) \rightarrow true$ **then**
11: **if** $policyList(P_{ID}) \rightarrow true$ **then**
12: $Result \leftarrow retrieveEHRs(PK, Addr)$
(refer to Algorithm 4)
13: break;
14: **else**
15: $Result \leftarrow penalty(PK, action)$
16: break;
17: **else**
18: $Result \leftarrow penalty(PK, action)$
19: break;
20: **end**
21: **end**
22: **return** *Result*;

We consider two use cases with two mobile devices which represent authorized user and unauthorized user, respectively. For the first case, a mobile user, i.e. a doctor, acts as a blockchain client to request data on cloud storage. As he is an e-health network participant, he know patient information in the blockchain network to request data of his interest. For the second case, a mobile user acts as an attacker to access illegally to the cloud storage. He is curious about the sensitive information of patient in the e-health system and wants to steal medical records on our cloud blockchain.

For experimental evaluations, we focus on two performance metrics: access control and network overheads. For the access control evaluation, we build a mobile cloud application on an Ethereum blockchain network. Mobile users (including authorized and unauthorized users) will join the blockchain network, make transactions to request data using a blockchain client platform on their Android phone. In this way, we investigate access control results for each type of data request. Access control analysis is achieved if the proposed scheme can allow authorized users to retrieve efficiently EHRs and detect any potential threats to data storage on cloud. The access control algorithm based on smart contract should be able to analyse and prevent any attackers to our e-health system, aiming to achieve a reliable EHRs sharing on blockchain.

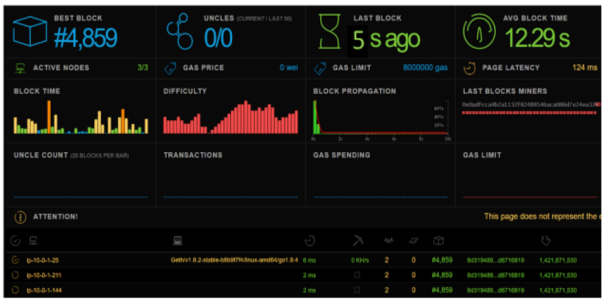

FIGURE 11. The running Ethereum blockchain for EHRs sharing on Amazon cloud.

In addition to access control evaluation, we also investigate network latency that is also an important indicator to show the effectiveness of real-time EHRs sharing system. We measure the time overhead of the user authentication process run by smart contract by using a Kinesis Data Analytics service on Amazon cloud. Besides, communication latency is also evaluated by analysing the time taken to send a data request and receive the requested data on the blockchain client. The evaluation results based on our testing methodology are presented in the next section.

VI. EXPERIMENTAL RESULTS

To implement our EHRs sharing framework, we first deployed a private Ethereum blockchain on AWS as illustrated in Fig. 11. Data access and transactions are recorded and shown on the web interface for monitoring. Based on blockchain settings, we deployed smart contracts, IPFS storage, established network entities and connected with mobile applications to build our e-health framework. With these settings, we operated the EHRs sharing system and evaluated the efficiency of our design through two main performance metrics: access control and network overheads.

A. ACCESS CONTROL PERFORMANCE

We present two use cases with authorized and unauthorized access to evaluate the performance of our EHRs sharing model with a designed access control (Fig. 12). The objective of our framework is to allow authorized entities (such as healthcare providers) to retrieve effectively EHRs on cloud, while being able to prevent unauthorized access to our EHRs resources. A mobile user such as a doctor, who wants to access EHRs of his patient on cloud, can use our mobile application with a mobile user interface to create an Ethereum account and register user information for interacting with the blockchain (Fig. 12(a)). After his request is verified by the cloud EHRs manager, he now starts to make a transaction to access EHRs by providing the address of his patient

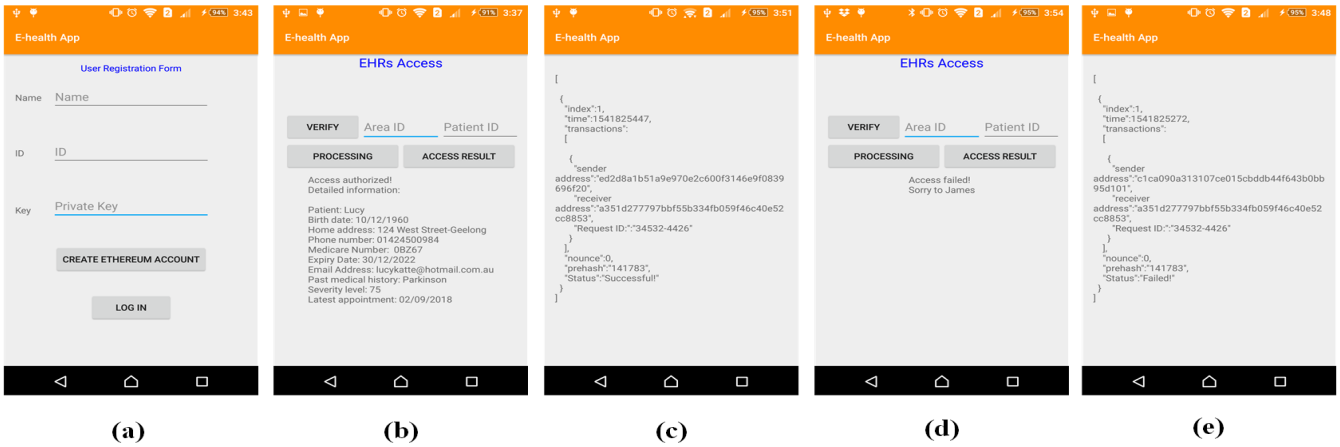

FIGURE 12. Illustrations of EHRs access results on Android phones: a) User Registration form with Ethereum account, b) EHRs access results of an authorized user, c) Transaction record of authorized EHRs access, d) EHRs result of an unauthorized user, and e) Transaction record of unauthorized EHRs access.

(including *AreaID* and *PatientID*) as shown in Fig. 12(b). Our EHRs system will then return data access results which are also updated on his mobile interface (Fig. 12(b)). Thus, the doctor can acquire all medical records of his patient to analyse and provide healthcare supports accordingly. Now the EHRs access process is finished and transaction is appended to blockchain by the cloud miner and broadcast to all entities in the network. Therefore, a patient can keep track of the exchange of their EHRs and know exactly who uses their data (Fig. 12(c)), thus ensuring data ownership of individuals and network trustworthiness.

In the case of unauthorized access, the smart contract will verify and detect by the access protocol with a predefined policy list. Such illegal request is prevented and discarded from our EHRs database, and a warning message is returned to the requester (see Fig. 12(d)). A corresponding transaction for unauthorized access is also issued by the smart contract (see Fig. 12(e)). Obviously, the use of blockchain can address effectively challenges mentioned in the literature in controlling patient information and monitoring user access, which can improve system reliability and data privacy.

With the analyzed access control results, it is clearly that our design achieves a secure data sharing among mobile users, with the capability of user identity, authentication. More importantly, the access control scheme enabled by smart contract design is capable of preserving sensitive medical data against external attacks. These performance results achieve the first design goal stated in the Section IV-D.

B. NETWORK LATENCY

We measured the average time consumption on cloud for processing many access requests simultaneously (Fig. 13(a)). The mechanism with user authentication based on smart contract consumes more time to process user requests, compared with the non-authenticated scheme. This overhead comes from time consumption for user identity and access authentication. However, in the worst case (seven requests),

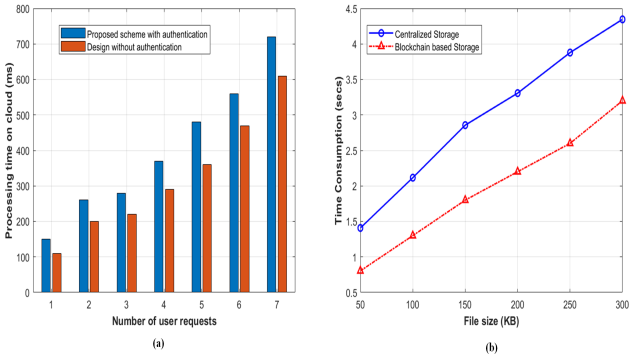

FIGURE 13. Time overheads: a) Processing time on cloud versus user requests and b) Time consumption for EHRs access on cloud storage.

the additional overhead is only about 100ms, which is still small and considered acceptable in real scenarios. This result highlights a lightweight design for user authentication and access control of our model.

We also evaluated the performance of our EHRs sharing system in terms of communication overhead for accessing EHRs on blockchain cloud storage IPFS (Fig. 13(b)). We measured time consumption for the EHRs access process from requesting data access to receiving data on the mobile phone. We also compared the time overhead of EHRs retrieval on our blockchain based storage IPFS with centralized storage enabled by AWS S3 service. The experiment results with respect to different EHRs file sizes show our decentralized storage design on blockchain outperforms the conventional scheme with centralized storage in terms of time overhead. This result also demonstrates the efficiency of the proposed EHRs sharing on cloud blockchain.

With network latency analysis, it is obvious that our design achieve a lightweight EHRs sharing with minimum system overheads, compared to conventional cloud design. This performance result achieves the second design goal stated in the Section IV-D.

TABLE 1. Comparison our system with other works.

Feature	[14]	[16]	[17]	[19]	[20]	Our system
Flexibility	N	N	N	N	Y	Y
Availability	N	N	N	Y	Y	Y
Decentralized access	N	N	Y	Y	Y	Y
Identity Management	N	Y	N	Y	N	Y
User authentication	Y	Y	Y	Y	N	Y
Integrity	Y	Y	Y	Y	Y	Y
Data privacy	Y	Y	Y	Y	Y	Y

VII. SECURITY ANALYSIS AND DISCUSSION

In this section, we provide performance analysis on security of the proposed system via two threat scenarios. Also, technical features of the designed data sharing scheme are also discussed to highlight the usefulness and feasibility of our design.

A. SECURITY ANALYSIS

Security are always a great concern in medical data sharing systems where sensitive patient information must be preserved well against potential threats to guarantee patient privacy and network security. We prove the security of our design through the following theorems.

Theorem 1: Assume that an attacker can access to the data storage system without consent of EHRs manager, such attacker may find it very difficult to retrieve and read medical data from our system.

Proof: In our design, all medical records are encrypted with the public key of EHRs manager for uploading to the decentralized cloud storage. To retrieve cloud data, any requestors need to know the private key of EHRs manager to decrypt such data package. Note that this private key is unique and only known by the EHRs manager. Therefore, it is very challenging for the attacker to guess the private key to decrypt and obtain data on cloud.

Theorem 2: Suppose that an adversary can send unauthorized transactions to request our cloud e-healthcare records. It is impossible for such adversary to tamper our access control framework to acquire access permission.

Proof: In order to request an authorized transaction, a mobile user needs their private key (SK), public key (PK) and a request ID, which are used to sign the transaction before sending to cloud storage as the syntax $AuthorizedTx \leftarrow Sign_{SK}(rawTx\{PK, requestID, timestamp\})$. It is noticed that only the user public key is visible to all entities in the blockchain network, while the private key is only known by its user. Therefore, the adversary is unable to have such private key to sign the transaction for accessing data. Further, any invalidated transactions will be removed by the miner from the blockchain network, which make our access control system resistant with external attacks.

B. DISCUSSION

The proposed EHRs sharing system is discussed and evaluated under various performance metrics to demonstrate the feasibility of our model for real usability scenarios.

1) FLEXIBILITY

Since our design is deployed on a mobile platform, any users with smartphones can easily to work on our system while allowing the freedom of users with high flexibility. Our system can work well with different mobile platforms, including Android and iOS versions, increasing the usability of our design in different healthcare system.

2) AVAILABILITY

Our system allows authorized mobile users to access e-healthcare records anytime and anywhere with a mobile application. The use of mobile app allows users to interact with our system in a real time and dynamic manner, with highly available medical data on cloud.

3) AVOID SINGLE POINT OF FAILURE

Our design employed the decentralized storage system IPFS that solves effectively the single point of failure problem. Besides, access control enabled by the blockchain technique is running in a peer-to-peer manner among decentralized entities that can also contribute to overcome this challenge.

4) INTEGRITY

Integrity guarantees that patient information is shared between authorized users without any change. Medical records collected from mobile gateways are always encrypted to avoid any alterations. Meanwhile, for EHRs sharing, mobile users are unable to modify the signed transactions to smart contracts and no any entities can tamper and change content of recorded transactions. Importantly, mobile users cannot have rights to change or alter the agreement in the smart contract and access policies in our scenario.

5) DATA PRIVACY

By exploiting security capability of blockchain and smart contracts, our access control scheme guarantees data privacy and data ownership of individuals. Malicious access are blocked by user identity capability and authorization of smart contract, preventing potential threats from accessing to our cloud storage. Moreover, illegal transactions will be invalidated and removed from our blockchain network by the consensus process. Another interesting feature of our design is that all entities in the blockchain share equal data management rights and monitor all transactions and messages. Therefore, any modification to cloud medical records can be detected easily by mobile users and informed to the cloud manager for safeguarding patient data privacy.

With a comprehensive analysis on various design aspects, our system can be preserved against potential attacks and threats. Our design also brings interesting features including flexibility, availability with high system integrity and data privacy, all of which can be useful to healthcare applications. These performance results achieve the last design goal stated in the Section IV-D.

Further, based on extensive discussions on literature studies in Section II, we show the comparison of our proposed design with other relevant works in *Table 1* in different technical features by using two options: Y-yes (available) and N-No (unavailable). The comparison results demonstrate that the proposed design outperforms conventional schemes and thus can provide a promising solution for improving current e-health applications.

C. EHRS SHARING CHALLENGES AND OPEN ISSUES

In addition to potentials of blockchain in EHRs sharing and e-healthcare systems, there are several other challenges and open issues that should be carefully addressed.

1) SCALABILITY

In modern e-health scenarios, there are many healthcare providers who want to participate in collaborative e-health services, i.e. collaborative healthcare between multiple hospitals. This requires a scalable and manageable IoT architecture which ensures reliable and robust interconnection of multiple mobile devices within the mobile cloud blockchain network [29]. Besides, different healthcare providers can have different roles based on their functionality, such as doctor, clinicians, physicians, nurses. Therefore, there is a need to extend the current EHRs sharing system, aiming to adapt to scalable e-health scenarios while guaranteeing high quality of user experience and e-health sharing reliability.

2) LOW NETWORK LATENCY

Network latency is a critical issue in real-time healthcare applications. For example, data streams generated from distributed wearable devices are high in volume and at fast rate, which can lead to traffic congestion on the cloud server. Further, due to the physical distance to mobile devices, conventional clouds based health services experience high network latency which may not be suitable to time-sensitive health applications. To overcome such challenges, mobile edge cloud can be a promising solution to provide nearby health services to mobile users with low network latency [30]. From the blockchain perspective, solutions for lightweight blockchain design in healthcare are necessary to optimize data processing and transaction communication for better latency efficiency [45].

3) DATA UPLOADING PRIVACY

Uploading medical data to cloud for sharing in healthcare blockchain can introduce some critical privacy issues that have been largely ignored by existing literature studies. For example, in cloud blockchain networks, the cloud server can curious about medical resources and steal sensitive patient data without consent of patients. More importantly, although the blockchain platform allows entities to keep track of their transaction records, curious miners can infer personal information such as user location or usage pattern information during the mining process. In such contexts, novel strategies which combine uploading optimization and user protection may be useful. Specially, uploading techniques using reinforcement learning (RL) have just emerged as a promising solution to solve such critical problems in healthcare [46].

4) QUALITY OF SERVICE (QOS)

To achieve high quality of e-health services, blockchain-based healthcare applications require QoS guarantees in terms of requirements such as data availability, flexibility and usability. In this regard, the improvement of current mobile e-health platform for both user and cloud server perspectives should be considered in e-health system design. Particularly, the integration of e-health system with intelligent services on blockchain can be an interesting topic. For instance, automatic clinical support platforms using data mining or machine learning can be integrated at the cloud to analyse medical records and predict health problems of patients in a dynamic manner. This can help healthcare providers, i.e. doctors to provide instant healthcare services, while data privacy and security are still guaranteed.

VIII. CONCLUSIONS

This paper proposes a novel EHRs sharing scheme enabled by mobile cloud computing and blockchain. We identify critical challenges of current EHRs sharing systems and propose efficient solutions to address these issues through a real prototype implementation. In this work, our focus is on designing a trustworthy access control mechanism based on a single smart contract to manage user access for ensuring efficient and secure EHRs sharing. To investigate the performance of the proposed approach, we deploy an Ethereum blockchain on the Amazon cloud, where medical entities can interact with the EHRs sharing system via a developed mobile Android application. We also integrate the peer-to-peer IPFS storage system with blockchain to achieve a decentralized data storage and data sharing. The implementation results show that our framework can allow medical users to share medical data over mobile cloud environments in a reliable and quick manner, in comparison to conventional schemes. In particular, our access control can identify and prevent effectively unauthorized access to the e-health system, aiming for achieving a desired level of patient privacy and network security. We also provide security analysis and extensive evaluations on various technical aspects of the proposed system, showing advantages of our proposal over existing solutions. Based on the merits of our model, we believe that our blockchain enabled solution is a step towards efficient management of e-health records on mobile clouds, which is promising in many healthcare applications.

REFERENCES

[1] T.-T. Kuo, H.-E. Kim, and L. Ohno-Machado, "Blockchain distributed ledger technologies for biomedical and health care applications," *J. Amer. Med. Inf. Assoc.*, vol. 24, no. 6, pp. 1211–1220, 2017.

[2] M. Mettler, "Blockchain technology in healthcare: The revolution starts here," in *Proc. 18th IEEE Int. Conf e-Health Net., Appl. Services*, Sep. 2016, pp. 1–3.

[3] W. J. Gordon and C. Catalini, "Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability," *Comput. Struct. Biotechnol J.*, vol. 16, pp. 224–230, 2018.

[4] 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.*, 2017, pp. 650–659.

[5] M. Hölbl, M. Kompara, A. Kamišalic, and L. N. Zlatolas, "A systematic review of the use of blockchain in healthcare," *Symmetry*, vol. 10, no. 10, p. 470, 2018.

[6] S. Jiang, J. Cao, H. Wu, Y. Yang, M. Ma, and J. He, "BlocHIE: A blockchain-based platform for healthcare information exchange," in *Proc. IEEE Int. Conf. Smart Comput. (SMARTCOMP)*, Jun. 2018, pp. 49–56.

[7] L. A. Tawalbeh, R. Mehmood, E. Benkhlifa, and H. Song, "Mobile cloud computing model and big data analysis for healthcare applications," *IEEE Access*, vol. 4, pp. 6171–6180, 2016.

[8] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, Jun. 2015.

[9] A. Bahga and V. K. Madisetti, "A cloud-based approach for interoperable electronic health records (EHRs)," *IEEE J. Biomed. Health Inform.*, vol. 17, no. 5, pp. 894–906, Sep. 2013.

[10] E. AbuKhousa, N. Mohamed, and J. Al-Jaroodi, "e-Health cloud: Opportunities and challenges," *Future Internet*, vol. 4, no. 3, pp. 621–645, 2012.

[11] M. Meingast, T. Roosta, and S. Sastry, "Security and privacy issues with health care information technology," in *Proc. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Aug./Sep. 2006, pp. 5453–5458.

[12] A. Ghazvini and Z. Shukur, "Security challenges and success factors of electronic healthcare system," *Procedia Technol.*, vol. 11, pp. 212–219, 2013.

[13] C. Esposito, A. De Santis, G. Tortora, H. Chang, and K.-K. R. Choo, "Blockchain: A panacea for healthcare cloud-based data security and privacy?" *IEEE Cloud Comput.*, vol. 5, no. 1, pp. 31–37, Jan./Feb. 2018.

[14] R. Wu, G.-J. Ahn, and H. Hu, "Secure sharing of electronic health records in clouds," in *Proc. 8th Int. Conf. Collaborative Comput., Netw., Appl. Worksharing (CollaborateCom)*, Oct. 2012, pp. 711–718.

[15] A. Ibrahim, B. Mahmood, and M. Singhal, "A secure framework for sharing electronic health records over clouds," in *Proc. IEEE Serious Games Appl. Health*, May 2016, pp. 1–8.

[16] Z. Ying, L. Wei, Q. Li, X. Liu, and J. Cui, "A lightweight policy preserving EHR sharing scheme in the cloud," *IEEE Access*, vol. 6, pp. 53698–53708, 2018.

[17] V. Ramani, T. Kumar, A. Bracken, M. Liyanage, and M. Ylianttila, "Secure and efficient data accessibility in blockchain based healthcare systems," in *Proc. GLOBECOM*, Dec. 2018, pp. 206–212.

[18] N. Rifi, E. Rachkidi, N. Agoulmine, and N. C. Taher, "Towards using blockchain technology for eHealth data access management," in *Proc. IEEE 4th Int. Conf. Adv. Biomed. Eng.*, Oct. 2017, pp. 1–4.

[19] Q. I. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, "MeDShare: Trust-less medical data sharing among cloud service providers via blockchain," *IEEE Access*, vol. 5, pp. 14757–14767, 2017.

[20] X. Liang, J. Zhao, S. Shetty, J. Liu, and D. Li, "Integrating blockchain for data sharing and collaboration in mobile healthcare applications," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commu. (PIMRC)*, Oct. 2017, pp. 1–5.

[21] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems," *IEEE Access*, vol. 6, pp. 38437–38450, Jun. 2018.

[22] M. Steichen, R. Norvill, B. F. Pontiveros, and W. Shbair, "Blockchain-based, decentralized access control for IPFS," in *Proc. IEEE Blockchain*, Jul. 2018, pp. 1499–1506.

[23] M. S. Ali, K. Dolui, and F. Antonelli, "IoT data privacy via blockchains and IPFS," in *Proc. 7th Int. Conf. Internet Things*, Oct. 2017, p. 14.

[24] Y. Chen, H. Li, K. Li, and J. Zhang, "An improved P2P file system scheme based on IPFS and blockchain," in *Proc. IEEE Big Data (Big Data)*, Dec. 2017, pp. 2652–2657.

[25] R. Ausanka-Crues. *Methods for Access Control: Advances and Limitations*. [Online]. Available: http://citeseerx.ist.psu.edu/?doi=10.1.1.596.2814

[26] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594–1605, Apr. 2019.

[27] Y. Zhang, D. He, and K.-K. R. Choo, "BaDS: Blockchain-based architecture for data sharing with ABS and CP-ABE in IoT," *Wireless Commun. Mobile Comput.*, vol. 2018, Oct. 2018, Art. no. 2783658.

[28] R. Xu, Y. Chen, E. Blasch, and G. Chen, "BlendCAC: A smart contract enabled decentralized capability-based access control mechanism for the IoT," *Computers*, vol. 7, no. 3, p. 39, 2018.

[29] O. Novo, "Blockchain meets IoT: An architecture for scalable access management in IoT," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1184–1195, Apr. 2018.

[30] J. Kang *et al.*, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet Things J.*, to be published.

[31] *Ethereum Blockchain App Platform*. Accessed: Mar. 25, 2018. [Online]. Available: https://www.ethereum.org

[32] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[33] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitco.in/pdf/bitcoin.pdf

[34] J. Benet. (2014). *IPFS-Content Addressed, Versioned, P2P File System*. [Online]. Available: https://arxiv.org/abs/1407.3561

[35] B. Confais, A. Lebre, and B. Parrein, "An object store service for a fog/edge computing infrastructure based on IPFS and a scale-out NAS," in *Proc. IEEE Fog Edge Comput. (ICFEC)*, May 2017, pp. 41–50.

[36] *Amazon Web Services (AWS)–Cloud Computing Services*. [Online]. Available: https://aws.amazon.com/

[37] *Solidity 0.5.3 Documentation*. [Online]. Available: https://solidity.readthedocs.io/en/develop/

[38] *Ethereum on Android*. [Online]. Available: https://github.com/ethereum/go-ethereum/wiki/Ethereum-on-Android

[39] *Lightweight Java and Android Library for Integration With Ethereum Clients*. [Online]. Available: https://github.com/web3j/web3j

[40] *Setting Up IPFS on AWS*. [Online]. Available: https://github.com/CromonMS/setup-ipfs-aws

[41] *BioKin*. [Online]. Available: http://biokin.com.au

[42] *Android App for the InterPlanetary File System*. [Online]. Available: https://github.com/ligi/IPFSDroid

[43] *Big File and String RSA Encryption by Android*. [Online]. Available: https://github.com/haodynasty/android-rsa

[44] *Decoder and Encoder for the Ethereum ABI*. [Online]. Available: https://github.com/ethereumjs/ethereumjs-abi

[45] Y. Liu, K. Wang, Y. Lin, and W. Xu, "LightChain: A lightweight blockchain system for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, to be published.

[46] M. Min *et al.*, "Learning-based privacy-aware offloading for healthcare IoT with energy harvesting," *IEEE Internet Things J.*, to be published.

DINH C. NGUYEN received the B.E. degree (Hons.) in electrical and electronic engineering from the Ho Chi Minh City University of Technology, Vietnam, in 2015. He is currently a Ph.D. scholar with the School of Engineering, Deakin University, Victoria, Australia. His research interests focus on security and privacy in the Internet of Things (IoT), mobile cloud computing, and blockchain. He is currently working on adopting blockchain for secure communication networks, including clouds and the IoT. He is a recipient of the Data61 PhD scholarship, CSIRO, Australia.

PUBUDU N. PATHIRANA was born in Matara, Sri Lanka, in 1970. He was educated at Royal College Colombo. He received the B.E. degree (Hons.) in electrical engineering, the B.Sc. degree in mathematics, in 1996, and the Ph.D. degree in electrical engineering, in 2000, from The University of Western Australia, all sponsored by the Government of Australia on EMSS and IPRS scholarships, respectively. He was a Postdoctoral Research Fellow with Oxford University, Oxford, a Research Fellow with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia, and a Consultant to the Defense Science and Technology Organization (DSTO), Australia, in 2002. He was a visiting Associate Professor with Yale University, in 2009. He is currently an Associate Professor with the School of Engineering, Deakin University, Geelong, Australia. His current research interests include bio-medical assistive device design, human motion capture, mobile/wireless networks, rehabilitation robotics, and radar array signal processing.

MING DING (M'12–SM'17) is currently a Senior Research Scientist with Data61 (previously known as NICTA), CSIRO, Australia. He has authored over 80 papers in the IEEE journals and conferences, all in recognized venues, and about 20 3GPP standardization contributions, as well as, a Springer book *Multi-point Cooperative Communication Systems: Theory and Applications*. He currently holds 14 US patents and has co-invented over 100 patents on 4G/5G technologies in CN, JP, and EU. He was the Lead Speaker of the industrial presentation on unmanned aerial vehicles in the IEEE Globecom 2017, which was awarded as the Most Attended Industry Program in the conference. Also, he was awarded as the Exemplary Reviewer of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, in 2017. He is currently an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.

ARUNA SENEVIRATNE is currently a Foundation Professor of telecommunications with the University of New South Wales, Australia, where he holds the Mahanakorn Chair of telecommunications. He has also worked at a number of other Universities in Australia, U.K., and France, and industrial organizations, including Muirhead, Standard Telecommunication Labs, Avaya Labs, and Telecom Australia (Telstra). In addition, he has held visiting appointments with INRIA, France, and has been awarded a number of fellowships, including one at the British Telecom and one at the Telecom Australia Research Labs. His current research interest is in physical analytics: technologies that enable applications to interact intelligently and securely with their environment in real time. Most recently, his team has been working on using these technologies in behavioral biometrics, optimizing the performance of wearables, and the IoT system verifications.