Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 1968960, 16 pages
https://doi.org/10.1155/2019/1968960

Research Article

A Fog Computing-Based Architecture for Medical Records Management

Cícero A. Silva, Gibeon S. Aquino Jr., Sávio R. M. Melo, and Dannylo J. B. Egídio

Department of Informatics and Applied Mathematics (DIMAp), Federal University of Rio Grande do Norte, Natal 1524, RN, Brazil

Correspondence should be addressed to Cícero A. Silva; cicero@ppgsc.ufrn.br

Received 17 October 2018; Revised 30 December 2018; Accepted 3 February 2019; Published 27 February 2019

Guest Editor: Jean-François Méhaut

The aging of the world's population and the growth in the number of people with chronic diseases have increased expenses with medical care. Thus, the use of technological solutions has been widely adopted in the medical field to improve the patients' health. In this context, approaches based on Cloud Computing have been used to store and process the information generated in these solutions. However, using Cloud can create delays that are intolerable for medical applications. Thus, the Fog Computing paradigm emerged as an alternative to overcome this problem, bringing computation and storage closer to the data sources. However, managing medical data stored in Fog is still a challenge. Moreover, characteristics of availability, performance, interoperability, and privacy need to be considered in approaches that aim to explore this problem. So, this article shows a software architecture based on Fog Computing and designed to facilitate the management of medical records. This architecture uses Blockchain concepts to provide the necessary privacy features and to allow Fog Nodes to carry out the authorization process in a distributed way. Finally, this paper describes a case study that evaluates the performance, privacy, and interoperability requirements of the proposed architecture in a home-centered healthcare scenario.

1. Introduction

The aging of the world's population and the growth of the number of people with chronic diseases have been a concern for medical care agencies [2–4]. These factors are the main reasons of the increase on expenditures with healthcare [5]. To get an idea, in 2015 was spent US$ 7.3 trillion with health around the world [6] and it was estimated that this amount will reach US$ 8.7 trillion in 2020 [7]. Thus, several technological solutions have been developed to improve the delivery of medical services [4], optimizing the patients' treatment process and helping to prevent more serious medical conditions. Thereby, healthcare applications and solutions using sensors have been developed for this purpose and have generated a great deal of information about patients' health on a daily basis.

In this context, the Cloud Computing paradigm has been used to store and process this information. However, Cloud-based solutions can create delays in health applications, which can result in the failure of medical systems and originate misdiagnosis [8, 9]. Thus, Fog Computing (Fog) has emerged as an alternative to using Cloud, bringing computing and storage capabilities closer to applications and data sources and consequently preventing delays [10–13]. In this way, Fog can be used in the development of more efficient technological solutions for the health field.

Despite the use of Fog infrastructure being a promising paradigm, there are still sparse approaches for coping with the problem of providing more efficient storage repositories in the Fog Layer [14]. Consequently, the management of medical data stored in the Fog is also a big challenge [15]. Furthermore, these types of data are critical, because the unavailability or delays of them can hamper the diagnosis and the physicians' decisions, therefore resulting in greater complications to the patients' health or even causing his death.

Privacy is also an important feature because medical records are intimate information about the patient's life [8, 16–18]. Therefore, the control of access to such data needs to be guaranteed, avoiding the unauthorized use of such information by malicious people.

In this perspective, approaches based on Blockchain have been used to provide privacy to similar scenarios.

Based on this aforementioned problem, this article proposes an approach that aims at performing the management of the patients' medical records using the Fog Computing paradigm. In the solution, Fog and Blockchain were combined to provide the requirements discussed previously. Thus, Fog Computing-based techniques were used to ensure availability and performance, and Blockchain-based strategies were used to provide the privacy required for the medical domain.

The remainder of this article is organized as follows: Section 2 describes some related works. Section 3 shows a Blockchain overview. Section 4 describes the proposed approach and its architecture. A case study evaluating three nonfunctional requirements of the software architecture is described in Section 5. Finally, Section 6 discusses the study's findings.

2. Related Works

In a previous work, we performed a state of the art review in which it was shown how the Fog Computing paradigm has been used in healthcare [1]. The article showed the types of health applications that have been developed using Fog, the groups of diseases addressed in these solutions, the kinds of research performed, the Fog characteristics that have been adopted, the motivations for the use of Fog in the area, and the challenges that need to be explored to improve the use of this paradigm in the health field. We also identified the types of sensors used in the solutions, the architectural style most commonly adopted, and the protocols and standards used in the analyzed works. Figure 1 shows an overview of the review's findings.

With the review, it was identified that the management of medical data stored in the Fog Layer is still an open question [15]. Security and privacy are important features for such an approach, and they are widely cited in [8, 9, 12, 15–18]. Storage capacity is also another challenge, as Fog has limited space when compared to the Cloud [19]. Thus, the Fog-based storage strategies have to cope with this limitation. Besides, the analyzed works mention the performance related to the manipulation of information in the applications as the main reason to use Fog Computing in the health field [10, 15, 20–26].

At the same time, the study [14] corroborates the findings related to the lack of a health data management approach in Fog, stating that there is no proposed solution that uses it as an infrastructure to provide more efficient storage repositories. In fact, most studies address solutions that exploit the processing capacity of Fog. Work [27] shows a new algorithm responsible for the dynamic scheduling of these types of tasks in vehicular networks. A new algorithm that divides the data into blocks and allocates processing resources for them is proposed in [28]. An architecture of service delegation and allocation of Fog resources based on service size, completion time, and capacity is proposed in [29]. On the other hand, the works do not propose solutions to improve the task of storage in Fog.

Considering the challenges presented in the current solutions, we have proposed a software architecture that addresses issues of availability, performance, interoperability, and privacy as an approach to provide better management of medical records. Our proposal differs from the works analyzed in the literature by focusing on the improvement of the storage of this data in the Fog Layer and the use of Blockchain to provide the necessary privacy.

3. Blockchain Overview

Blockchain is a database ledger that stores transaction records between a peer-to-peer network in a decentralized, distributed, immutable, and secure manner [30, 31]. The original definition of Blockchain's function was created in 2008 by Satoshi Nakamoto in work [32]. In that article, Nakamoto used the Blockchain as a basis for the operation of the Bitcoin cryptocurrency.

Figure 2 shows the structure of a Blockchain database. It is formed by a set of chained blocks [33–35], hence the name Blockchain. Each block contains a set of actions generated by network participants, the so-called transactions [36–38]. These transactions are validated by Miners, networked computers that solve mathematical problems to obtain the right to create a new block [36, 39, 40].

Another important concept that relates to Blockchain is that of digital wallets [41], which are applications capable of sending, receiving, and maintaining transactions history performed by a network participant. If a transaction is sent from Wallet A to Wallet B, it is first sent to the network. After that, a Miner validates it and places it in a block. Subsequently, the block is sent to all nodes belonging to the network, which approve it. That block is chained to the existing blocks in Blockchain. Finally, the transaction is received by Wallet B. Those operations are shown in Figure 3.

A scheme of public key cryptography and digital signatures is used to identify the managed accounts in the digital wallets and to ensure the authorization of the transactions. Each transaction is signed by the source node to authorize the sending of data and their integrity is verified by cryptographic techniques [35].

4. Proposed Approach

This section presents the proposed approach, which was designed to provide better management of the patients' records. This approach aims to improve system availability and performance by storing a subset of the patients' information closer to the applications and data sources. Furthermore, the architecture ensures the safety of the patients' data through the use of access control techniques. It also ensures the privacy of information by unlinking the patient's identity from the information related to him/her.

4.1. Requirements. The proposed solution seeks to manage the patients' records adopting the strategy of enabling the patients to be able to manage their own health data. Thus, the

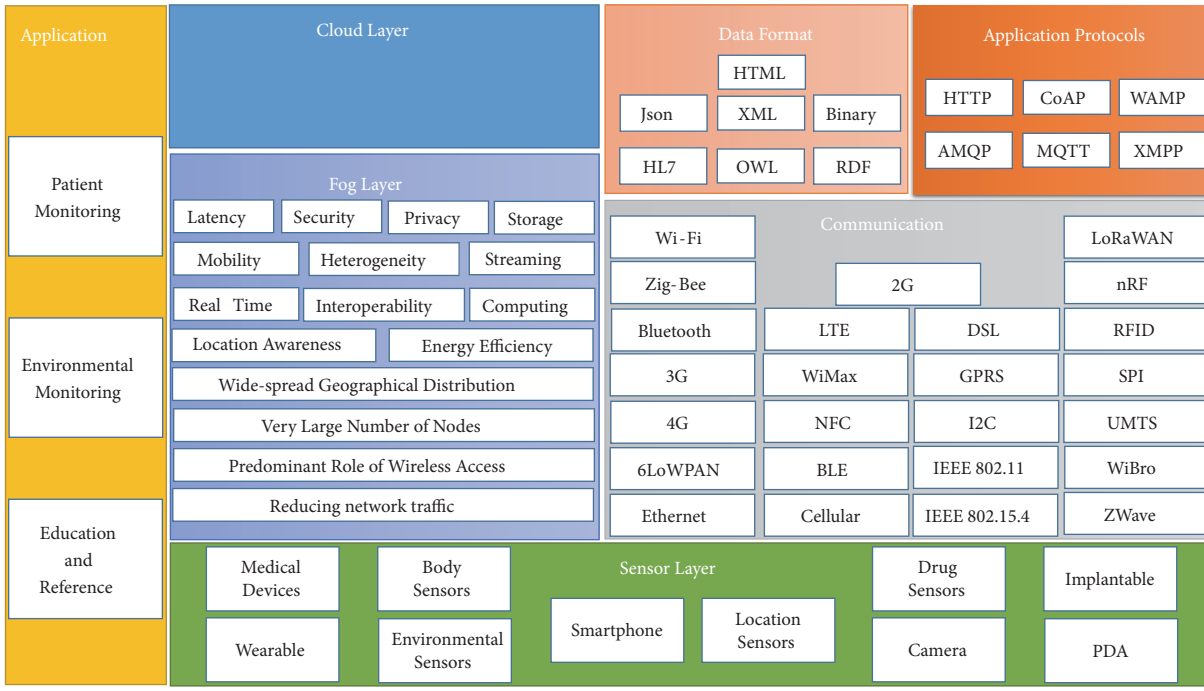


Figure 1: Technological view [1].

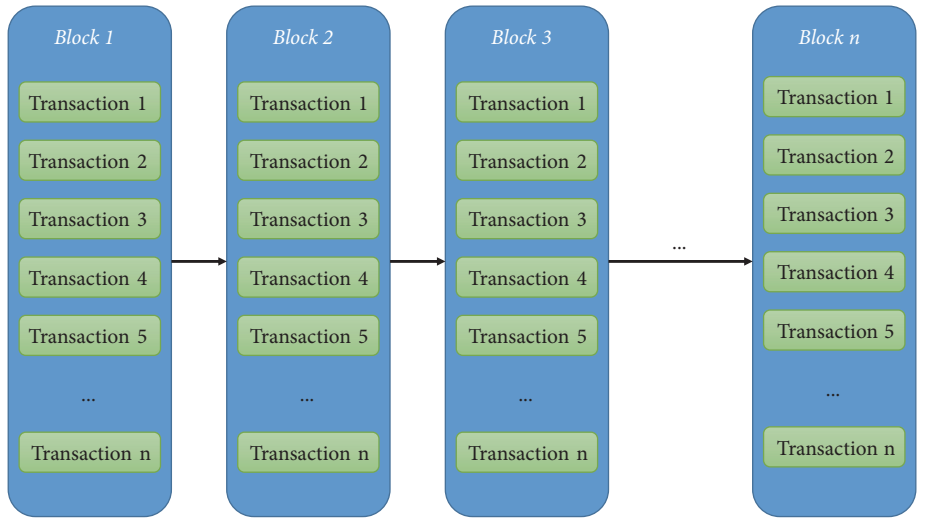


Figure 2: Blockchain structure.

following functional requirements (FR) have been identified to achieve this objective:

(i) *FR1, register patient application*: refers to the ability of the application to register itself in order to manage the patient's data;

(ii) *FR2, request data access*: refers to the ability of an application to request authorization to manipulate a subset of data from a particular patient;

(iii) *FR3, grant access to data*: refers to the patient's ability to control applications which may manipulate a subset of his data;

(iv) *FR4, view data in the medical record*: refers to the ability of the application to access patient-related medical information;

(v) *FR5, handle patient data*: refers to the ability of a previously authorized system to manipulate a subset of the patient's data.

In addition, the nonfunctional requirements (NFR) considered in the solution were as follows:

(i) *NFR1, availability*: refers to the software's tendency to be ready to perform its task when needed [42]. This is an extremely critical requirement in the medical field,

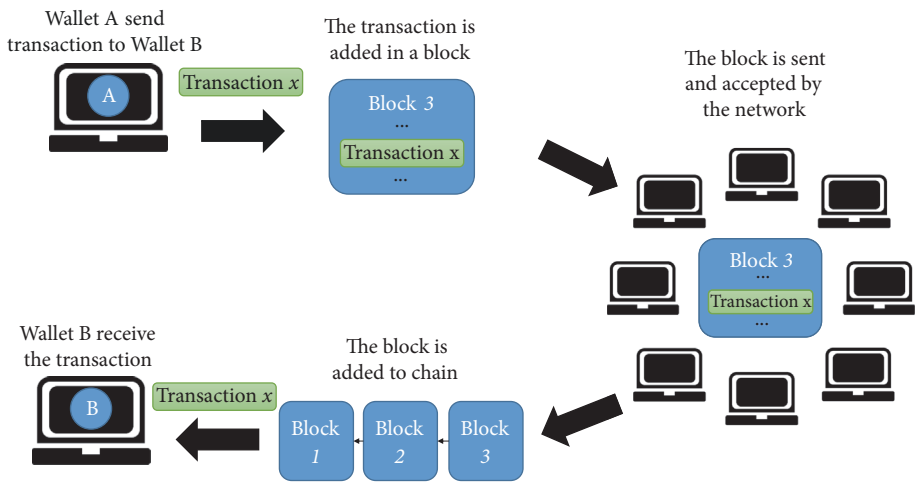


Figure 3: General operation of a Blockchain.

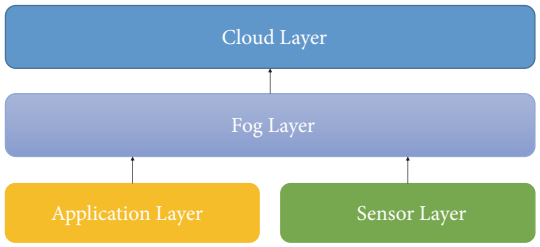


Figure 4: Architecture's layered view.

because the unavailability of information may result in further complications in the patient's health, which can lead him/her to death;

(ii) *NFR2, privacy*: refers to the software's ability to provide to an entity the option to limit the access of others to his/her personal information [43]. Medical data is characterized by being very critical information because it contains sensitive information for each individual. In this way, it is indispensable that the privacy of these information is guaranteed. Moreover, there are different environments and professionals responsible for patient care. Thus, it is important to guarantee the limitation of access to patient information according to the environment and the professional that is manipulating it;

(iii) *NFR3, performance*: refers to the software's ability to meet time requirements [42]. Rapid access to medical records is essential because the time factor is important in the preparation of diagnoses and treatments that prevent more serious medical conditions;

(iv) *NFR4, interoperability*: refers to the degree to which two or more systems can exchange meaningful information through interfaces in a particular context [42]. The medical area is characterized by having several environments with different systems, where information is generated in different formats. In certain situations, it is important that information created in one application can be used by others which are only able to manipulate a different format. Thereby, it is essential that the data generated can be interoperable between different medical systems.

4.2. Architectural Design. This section describes the designed approach, serving as a communication tool between stakeholders (analysts, architects, and programmers) and as the basis for analyzing and building a solution for the management of the patients' records. Thus, the description of this solution was based on the guidelines described in [42, 44].

This way, a set of views about it was generated, in which a view is a representation of a collection of elements of the system and the relations between them [42]. Also, the behaviors designed for the approach have also been described.

4.2.1. Layered View. Figure 4 shows the layered view of the proposed architecture. This type of view helps to bring the modifiability and portability attributes to the approach that is being defined [44]. In the elaboration of this view, the layers identified in the state of the art review mentioned in Section 2 and in the technological view shown in Figure 1 were used. Thus, it is composed of four layers, which are the following:

(i) *Sensor Layer*: responsible for monitoring patients through sensing devices;

(ii) *Application Layer*: responsible for accessing and manipulating the data from the patient's records and for controlling their use;

(iii) *Fog Layer*: responsible for managing and storing a subset of data from the medical records closer to the applications. Also, this layer is responsible for receiving data generated by the *Sensor Layer* devices and by the *Application Layer* systems. It also validates the access to this data subset;

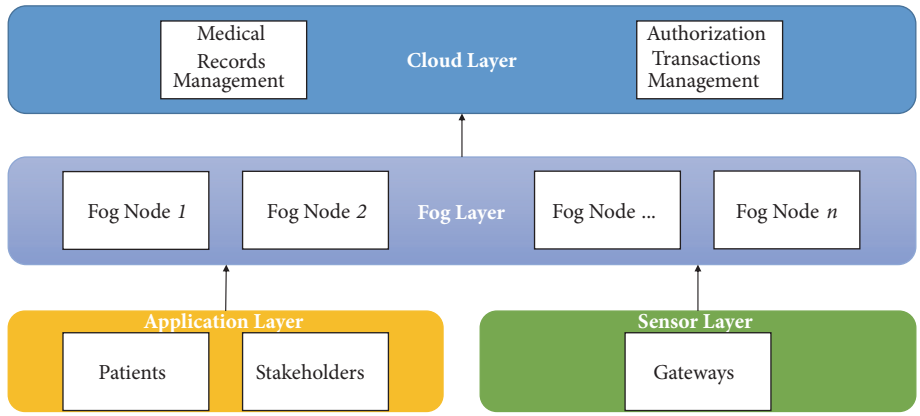


Figure 5: Architecture's layer decomposition view.

(iv) *Cloud Layer*: responsible for storing the complete set of patients' records, the proprietary data application information, and the records of authorization to access them.

Interacting with the *Sensor Layer*, there are various types of devices identified in the green part of Figure 1. These devices monitor patients, generating data for their medical records. Relating to the *Application Layer*, there are systems that manipulate the data from the patient's record and are used by patients, physicians, nurses, family members, rescue workers, hospital staff, and others.

4.2.2. Layer Decomposition View. An objective decomposition view shows how the responsibilities of the architecture are divided between the modules and how these modules are decomposed into submodules [44]. A decomposition view of the described layers is shown in Figure 5. At this stage of the project, Blockchain-based strategies were used to provide the necessary security and privacy requirements for this environment.

The *Gateway* module is a software component that receives the data generated on the sensors and sends them to the *Fog Layer*. Thus, this is one of the modules that implement the *FR2* and *FR5* requirements.

The *Application Layer* is composed of two submodules. The first one is the *Patient* module, which represents a software component that allows the patient to view the data related to his/her medical records and through which he/she can manage it, controlling which applications can manipulate the data and the subset of information each application can use. The *Patient* module functions as a Blockchain wallet. As such, it performs its registration and grants access to the managed data through the creation of transactions. This way, this module provides compliance with the *FR1*, *FR3,* and *FR4* requirements.

The *Stakeholder* is the second module of the *Application Layer*. It represents a software component that is responsible for requesting access to manipulate a subset of the patient's medical record data and uses that information as needed. Therefore, this component implements the *FR2* and *FR5* requirements. The *Stakeholder* module functions as a Blockchain wallet and performs access requests through transactions. Besides, this component is in charge of sending the changes made in the patient's data to the *Fog Layer*.

The *Fog Layer* is composed of a set of modules called *Fog Nodes*. These modules are software components that manage a subset of data from the patient's records to bring them closer to the applications. Thus, they provide the *NFR1* and *NFR3* requirements.

Also, these components register the *Patients* modules that manage a given subset of data and are in charge of the authorization validation process so that *Stakeholders* and *Gateways* can manipulate data from a patient. This way, *Fog Nodes* meet the *NFR2* requirements.

A *Fog Node* functions as a Blockchain Miner to validate these operations (transactions). It is also responsible for receiving the data generated in *Stakeholders* and *Gateways*. To do this, it provides a Representational State Transfer (REST) interface, providing access through POST, GET, PUT, and DELETE operations, capacity that allows the attendance of *NFR4*. Finally, it synchronizes the subset of data and authorizations with the *Cloud Layer*.

The *Medical Records Management* and the *Authorization Transactions Management* are the modules that make up the *Cloud Layer*. The first one is a software component whose function is to store the entire set of patient information in a relational database. Also, this module is responsible for receiving new data from the *Fog Layer* and making the access to the data that it stores available to that layer. Thus, it also provides a REST interface, granting access for data manipulation.

The *Authorization Transactions Management* module, in its turn, is a software component responsible for storing the entire set of records from the *Patients* applications that manage data, as well as the set of data access authorizations validated in the *Fog Nodes*. This way, this component functions as the *Blockchain* database, storing proprietary application registration transactions and authorization transactions. It is also another component that contributes to the fulfillment of the *NFR2* requirements.

A second view of the decomposition of the architecture is shown in Figure 6. In the image, a greater detailing of the

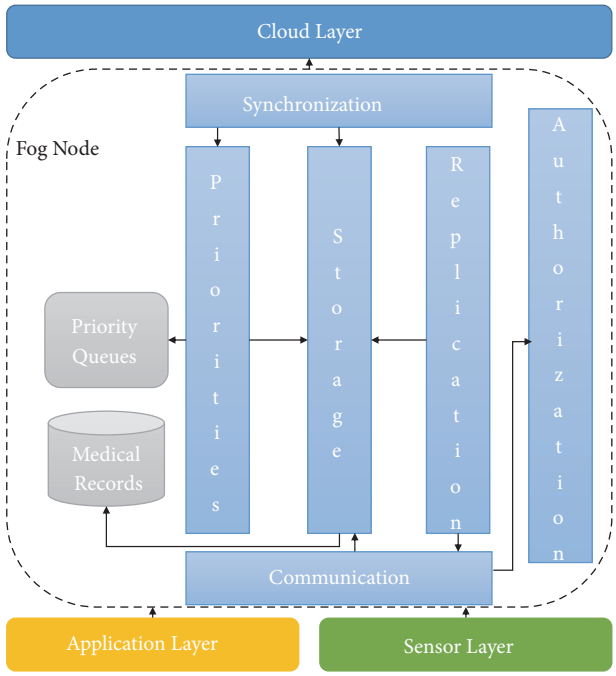


Figure 6: *Fog node's* decomposition view.

Fog Layer is provided, thus allowing the visualization of the decomposition of a *Fog Node*. As can be seen, each module of this type is composed of six submodules, which are the following:

(i) *Communication*: it is a software component in charge of registering applications that are connected to the *Fog Node*. Also, this module provides a REST interface for exchanging medical records with the *Sensor Layer* and the *Application Layer*. Besides, it provides an interface so that the *Sensor Layer* and the *Application Layer* can send transactions to the *Fog Layer*. Finally, this component also provides an interface that uses the *BitTorrent* protocol in the exchange of information between the *Fog Node* and the other modules that form the *Fog Layer*.

(ii) *Authorization*: it is a software component responsible for validating the registration of *Patient*-type modules. It also validates the authorizations for the *Stakeholders* and *Gateways* to manipulate patient data. Thus, this module acts as a Miner to perform these validations. This way, it validates whether an application that is trying to change a subset of data is allowed to do so. This is the component of the *Fog Node* responsible for meeting the *NFR2* requirements.

(iii) *Storage*: it is a software component responsible for storing a subset of the patient's record data in a relational database. Thus, this submodule helps to provide the *NFR1* and *NFR3* requirements.

(iv) *Replication*: it is a software component responsible for replicating to a nearby *Fog Node* a subset of new data stored on the *Fog Node* of which it is part. Thus, this submodule contributes to data availability as a fault tolerance technique, implemented through data redundancy. Finally, this submodule also helps to provide the *NFR1* requirement.

(v) *Priorities*: it is a software component that manages information related to the priority of the data stored in the *Fog Node*. Thus, this component keeps information regarding the criticality of the data, when it was last used and its storage time. This information is used to decide which data should be released first from *Fog Layer*, which should be kept stored at that layer, and which ones have the highest priority related to their availability.

(vi) *Synchronization*: it is a software component responsible for synchronizing the data stored in the *Fog Layer* with the data stored in the *Cloud Layer*. This way, this submodule is in charge of sending data from time to time to the *Cloud Layer* to keep it updated. It also looks for the information required by the *Application Layer* that is not stored in the *Fog Node*. Also, the Synchronization is in charge of freeing up storage space in the Fog Node, if necessary, using the *Priorities* information. Finally, it performs a prior search of the data that will be used in the applications, providing a greater performance in the delivery of this information.

4.2.3. Data Model View. The data model view describes the structure of the data used in the system as entities and relationships. This type of view helps guide the implementation phase and improves modifiability in data-centric systems [44]. Figure 7 shows the data model view of the proposed architecture, which is composed of the following entities:

(i) *ManagementApplication*: it represents an application that manages a patient's medical record. This entity helps to provide privacy for the architecture since the managed data will be associated with an application instead of with a person. This way, after a set of data is stored, it will not be possible to associate it with a person since no information will be available to allow this. Thus, this approach helps to meet the *NFR2* requirement.

(ii) *Patient*: it represents a patient who has his/her set of medical records stored. This entity contains basic information about a patient, such as weight, height, date of birth, etc. On the other hand, it will not include any information that allows the association of the data to a person, thus contributing to the privacy of the solution - and to the provision of the *NFR2* requirement. This way, the entity will not store attributes such as name and ID number.

(iii) *MedicalRecord*: it represents a medical data stored in a patient's medical record. As illustrated in the dashed part of the Figure 7, several generalizations can be created from this entity to represent more specific data such as examination and surgical procedure.

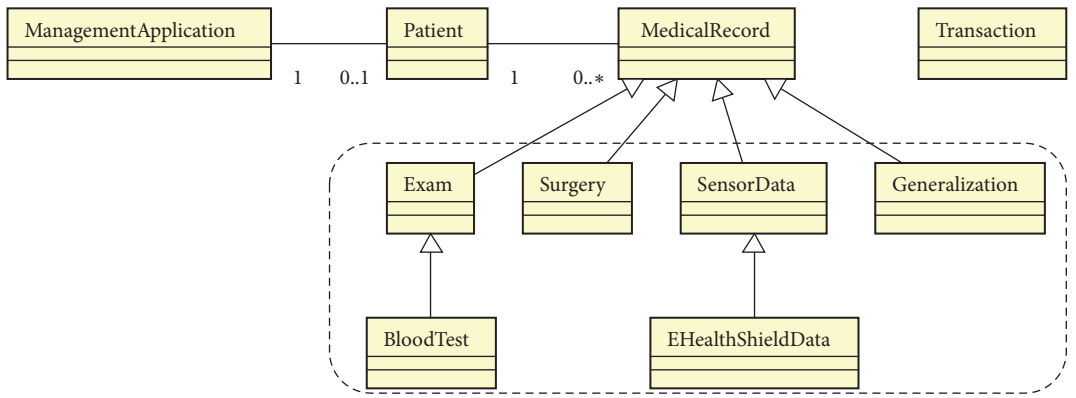


Figure 7: Architecture's data model view.

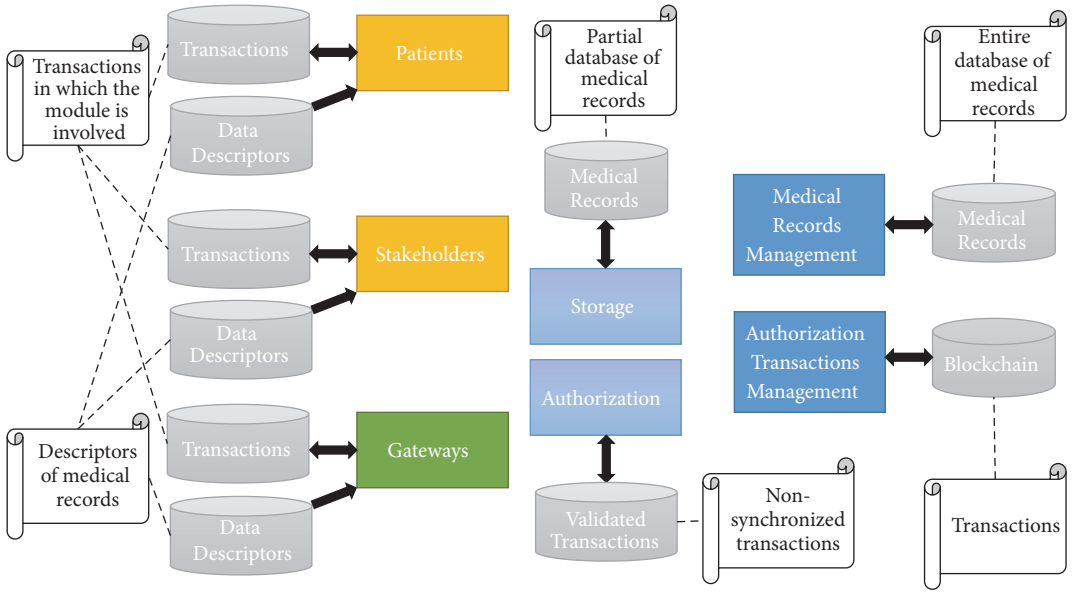


Figure 8: Architecture's repositories view.

(iv) *Transaction*: it represents the transactions that allow the management of a medical record. The types of transactions defined for the architecture were as follows:

(a) *Register patient application*: it is the transaction that provides the operation of registering an application to manage a medical record. Thus, this transaction helps to meet the *FR1*.

(b) *Request data access*: it is the transaction that provides the ability of an application to request access to manipulate a patient's data. Thus, this transaction helps meet the *FR2*.

(c) *Grant data access*: it is the transaction that provides the ability of a patient application to grant access for another application to manipulate a subset of data managed by it. This way, it helps the *FR3* to be met.

4.2.4. Repositories View. A repositories view displays one or more components called repositories, which contain extensive collections of persisted data. It also shows the components that read and write data in these repositories [44].

Figure 8 shows the architecture's repositories view. In the image, it is possible to see that the *Patient*, the *Stakeholder*, and the *Gateway* components have two repositories: the *Transactions* repository, which stores all the transactions to which the module is associated and the *Data Descriptors*, which persists the entire set of descriptors of the data types stored in a medical record.

The *Storage* component interacts with the *medical records* repository, which stores a subset of medical records data. The *Validated Transactions* repository is used by the *Authorization* component and stores the set of Validated Transactions that have not yet been added to the *Blockchain*.

The *Medical Records Management* component interacts with the *medical records* repository, which stores the entire

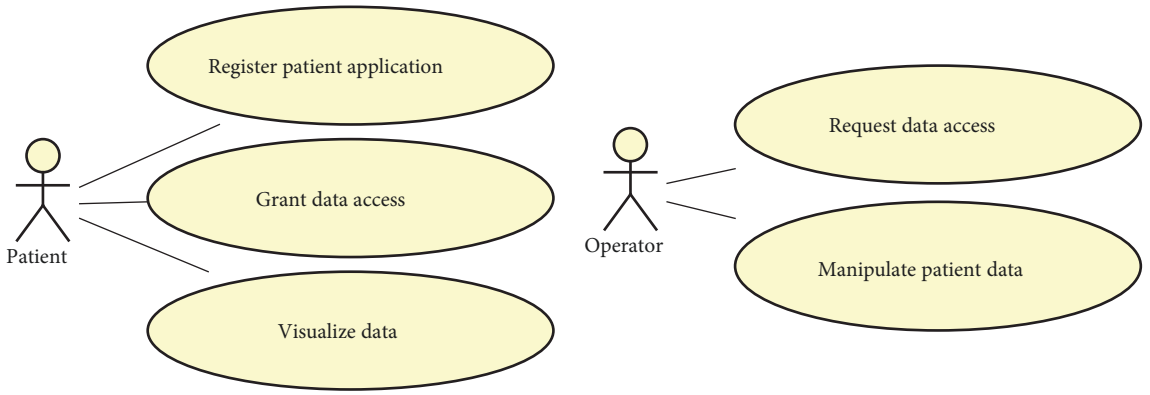


Figure 9: Architecture's use case diagram.

set of patient's medical records. Finally, the *Authorization Transactions Management* component interacts with the *Blockchain*, which stores all the transactions validated and added to the blocks chain.

4.3. Behaviors. The behavior documentation shows how the architectural elements interact by complementing the description of the architecture's views. In turn, this information provides many benefits during the development of the architecture and in the systems' maintenance phase [44].

Designing a use case diagram is a way of documenting behavior in architecture. This type of diagram helps in visualizing the functional requirements of a system and shows how they can be used by actors [45]. Figure 9 shows the use case diagram of the proposed architecture.

As shown in the figure, the diagram presents a use case for each functional requirement described in Section 4.1. Thus, the identified use cases were as follows:

(i) *Register patient application*: this use case begins when a patient starts to use a *patient* application. The first task he/she needs to do is to register his/her application so that he/she can manage his/her medical record.

(ii) *Request data access*: this use case starts when an operator of a *Stakeholder* application or a *Gateway* requests access to manipulate a patient's medical record.

(iii) *Grant data access*: this use case begins when a patient receives a notification in their *patient* application informing him/her that a particular module wants to use his/her medical record. Thus, it grants authorization for the requesting module to manipulate a subset of his/her data.

(iv) *Visualize data*: this use case starts when a patient selects a subset of data from his/her record to be viewed through his/her *patient* application.

(v) *Manipulate patient data*: this use case begins when a *Stakeholder* application or a *Gateway* needs to manipulate a subset of medical record data from a patient that previously authorized it.

The architecture proposed in this work has two operations to meet the use cases described above, which are as follows:

(i) *Transaction*: represents the transactions required for the registration and authorization process in the architecture. A *Transaction* consists of six parts:

  (a) *From*: the identifier of the module that created the transaction, i.e., the source;

  (b) *To*: the identifier of the module that received the transaction, i.e., the destination;

  (c) *Permissions*: set of addresses and operations to manipulate them:

    (1) *Address*: address of the subset of data to be manipulated, that is, Uniform Resource Identifiers (URIs) for data access;

    (2) *Operations*: operation(s) to be performed on the subset of data, that is, operations defined in the REST style;

  (d) *Type*: represents one of the three types that a transaction can assume, which are:

    (1) *RPA*: represents a transaction for the *Register patient application* behavior;

    (2) *RDA*: represents a transaction for the *Request data access* behavior;

    (3) *GDA*: represents a transaction for the *Grant data access* behavior.

  (e) *Expires in*: indicates the expiration date for the transaction. A Blockchain does not allow deletion of a transaction after it has been added to it. Thus, this item was used so that the patient indicates the maximum period of time that an application can use the requested subset of data.

  (f) *Others*: field for adding extra information about the transaction.

(ii) *Message*: represents the operation to use the data from medical records. A message is made up of five parts:

  (a) *Operation*: represents the REST operation to be performed on the data;

(b) *Address*: represents the URI of the data to be used;

(c) *Requester*: represents the identifier of the component that wants to use the data;

(d) *Transaction*: represents the identifier of an *RPA* transaction, if the data owner is willing to use it, or the identifier of a *GDA* transaction, if a component authorized to use the data is willing to use it;

(e) *Data*: represents the data to be sent in the message.

5. Case Study

This section describes a case study in which medical records of one patient, generated in a home-centered healthcare scenario, are managed through the approach proposed in this work. Particularly, the main objective of this case study is to evaluate the fulfillment of the nonfunctional requirements of performance, privacy, and interoperability, defined by the proposed approach. Finally, the planning and description of this case study followed the guidelines established in [46–48].

5.1. Planning. This case study instantiates the architecture, described in Section 4.2, to manage data of one patient in a home-centered healthcare scenario. Thus, it was intended to collect data about patient's health conditions through sensors, send them to the Fog through a Gateway, manage them through the proposed architecture, and make them available to be accessed by an authorized application. That application is represented by an information system, which is used by a nurse to check the health conditions of the person being treated. An overview of this scenario is shown in Figure 10.

The scenario of home-centric healthcare was chosen because it represents a paradigm change on patient care provided by the Internet of Things since it allows patients to continue being accompanied by health professionals while they are in their house. In this way, the treatment becomes more humanized, because it happens in the comfort of their residence and close to their family. At the same time, it reduces the occupation in the hospital environment.

In order to achieve the objective of this case study, we investigated the following research questions (RQ):

(i) *RQ1*: does the use of Fog Computing improve the access time to the patient's medical records?

(ii) *RQ2*: does the Blockchain-based privacy control strategy impair the access time to the patient's medical records?

(iii) *RQ3*: does the proposed approach allow the patient to restrict third-party access to his/her medical records?

(iv) *RQ4*: does the proposed approach allow different systems to exchange information?

The *unit of analysis* of this case study is the implementation of the architecture proposed applied to the scenario shown in Figure 10. Particularly, the performance, the privacy, and interoperability characteristics provided by the architecture were evaluated. So, the following data were collected:

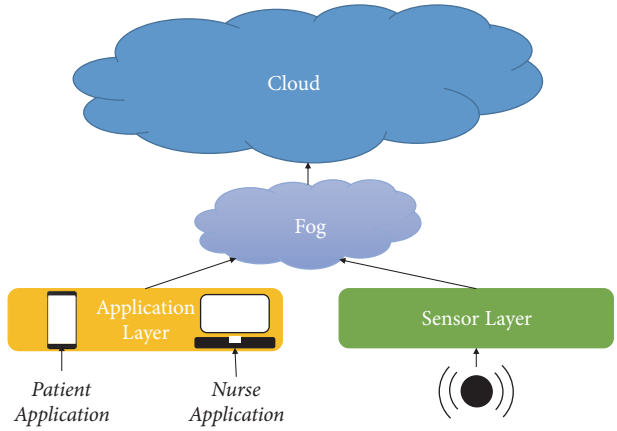


Figure 10: Home-centered healthcare scenario analyzed in the case study.

(i) The average time for accessing medical records generated in the home-centered healthcare scenario, without the use of Fog Computing

(ii) The average time for accessing medical records generated in the home-centered healthcare scenario, with the use of Fog Computing

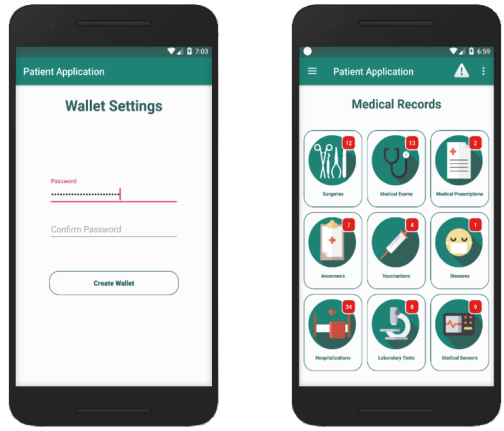
(iii) The average time for accessing medical records generated in the home-centered healthcare scenario, with the use of Fog Computing and the Blockchain-based privacy strategy

(iv) Interoperability demonstration between Gateway and applications

(v) Restriction demonstration of the nurse for accessing patient's medical records

5.2. Execution. Firstly, the Medical Records Management component was developed to enable the complete storage of data generated in the home-centered healthcare scenario. This component was implemented in the Java programming language and was hosted in a Cloud service. The Cloud is used to make part of Platform as a Service (PaaS) category and allows the publication of applications with a minimum configuration. It is also responsible for taking care of the environment the software performs, including security, operating system, and hardware responsibilities. The used service has different scalability characteristics, adjusting to support request peaks. Finally, this service performs instance scheduling automatically.

Subsequently, the development of a Fog Node with its respective submodules, shown in Figure 6, was started. This module was implemented on Java programming language and it is in charge of maintaining a subset of the medical records closest to the applications. Fog Node also runs a Blockchain Ethereum mining node, which validates the three types of transactions described in Section 4.2.3.



(a) (b)

Figure 11: Patient application (icons made by Freepik from www.flaticon.com).

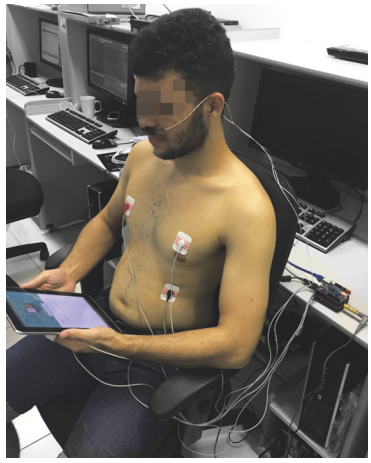


Figure 12: E-health shield collecting the patient's data.

The Ethereum Harmony (https://github.com/ether-camp/ethereum-harmony)implementation was used to create this node. It was chosen because it was developed in Java programming language, allowing the creation of private networks and for providing a JSON-RPC 2.0 (https://www.jsonrpc.org/specification) interface, which allows applications and the Gateway to communicate with Ethereum. After that, Fog Node was installed on a *Windows 10* machine with *AMD Phenom (tm) II X4 B97 3.20 GHz* processor and *5.5 GB of RAM* reserved for running it.

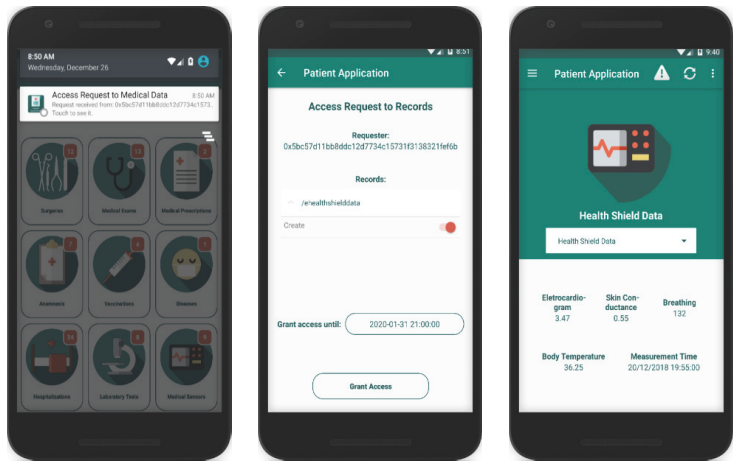
After that, we used the Android platform to develop the patient application. This application is in charge of managing access to the medical records generated in the scenario. The application register is the first step for a patient to use it, as shown in Figure 11(a). In this process, the application sends an *RPA*-type transaction to the Fog Node. Then, the patient starts to manage the set of medical records, as shown in Figure 11(b).

The next step was the development of the Gateway that is responsible for receiving the data obtained by monitoring the patient and sending them to the Fog Node. In this case study, an E-Health Shield (https://www.cooking-hacks.com/documentation/tutorials/ehealth-bio-metric-sensor-platform-arduino-raspberry-pi-medical) was used to monitor the patient, as shown in Figure 12. This type of device allows the collection of a person's body data and allows them to be sent to the Gateway. Table 1 shows the body information collected by E-Health Shield and used in this case study.

The software running on the Gateway was implemented in Java programming language. After, it was installed on a Raspberry Pi 2 model B. The first step in entering the data collected in the patient's medical records is to send a transaction requesting permission for this operation. Fog Node receives the transaction, validates it, and sends it to the patient application. The patient is notified that the Gateway is requesting access, as shown in Figure 13(a). The patient can then grant the permission, as shown in Figure 13(b). At this point, the Gateway begins sending the data collected by E-Health Shield to Fog Node and the patient can access them, as shown in Figure 13(c).

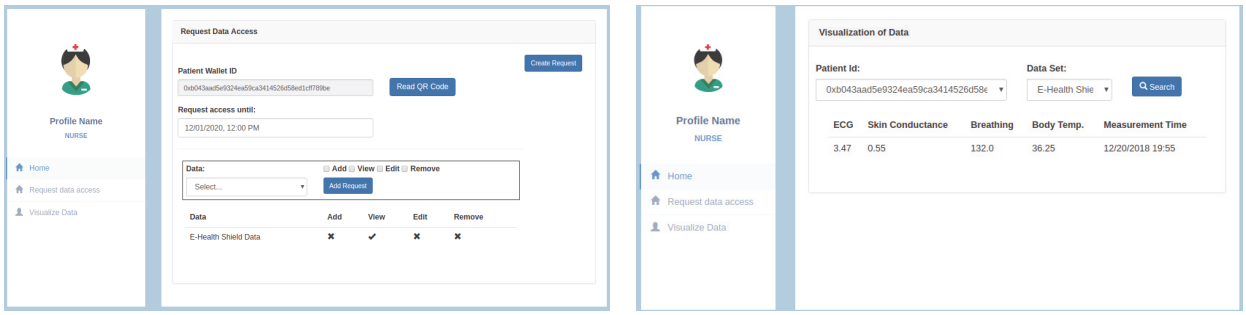
Subsequently, the Nurse Application was implemented in Java Web with the Spring Boot (https://spring.io/projects/spring-boot) framework. This application is used by the nurse to monitor the data collected about the patient's health. Firstly, it is necessary to request access to the data collected in the projected scenario to use it, as shown in Figure 14(a). The Nurse Application sends an *RDA*-type transaction to the Fog Node. That module validates the transaction and sends it to the patient application. The patient receives the notification and grants access to the nurse's application, as shown in Figure 15. At this moment, the nurse can follow the information collected by E-Health Shield, as shown in Figure 14(b).

Patient Application, Gateway, and Nurse Application make use of the Web3j (https://github.com/web3j/web3j)



(a) (b) (c)

Figure 13: Patient application granting access to the Gateway (icons made by Freepik from www.flaticon.com).



(a) (b)

Figure 14: Nurse application (Female Medical Nurse Flat Icon Vector.svg from Wikimedia Commons. Author: Videoplasty.com. License: Creative Commons Attribution-ShareAlike 4.0).

library to communicate with Blockchain Ethereum. This library is a Java implementation of a client for the JSON-RPC interface provided by Ethereum. Thus, applications and gateways can send and receive transactions with it.

After the implementation of all the software components necessary to the operation of the scenario, we set out to run tests with the Apache JMeter (https://jmeter.apache.org/) tool. Each test was repeated 100 times and in each one of them was configured to simulate a set of applications (1, 10, and 100 applications) accessing at the same time the data generated in the implemented scenario. In each access, each application performs 100 requests to the patient's data. Finally, we collected the average access time in each test.

The first environment tested used only Cloud to store the data generated in the scenario. In this way, applications and the Gateway communicated directly with the Cloud to search and create data. Table 2 shows the results obtained for this environment.

The second environment, besides using Cloud, added the Fog Layer to store a subset of the information generated in the scenario. So, the applications and the Gateway communicated with a Fog Node, which communicated with the Cloud. The results obtained for this environment are shown in Table 3.

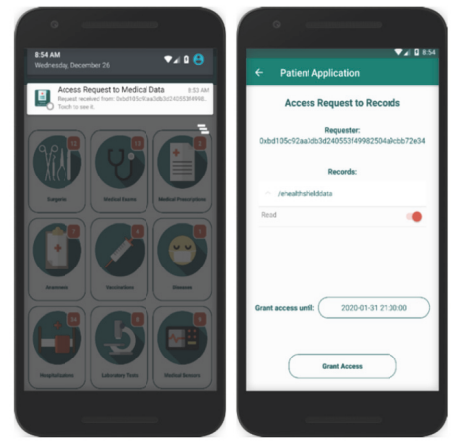


Figure 15: Patient application granting access to the Nurse Application (icons made by Freepik from www.flaticon.com).

In the latter environment, we added the Blockchain-based privacy control strategy in the Fog Layer. In this perspective, applications and the Gateway continue to communicate with

Table 1: Data collected by E-Health Shield.

Data	What is it used for?
Electrocardiogram	It is used routinely to evaluate the electrical and muscular functions of the heart.
Breathing	It is used to identify physiological instabilities.
Body temperature	It is used to identify the rise of diseases and the efficiency of treatments.
Skin conductance	It is used as an identifier of psychological or physiological arousal.

Table 2: Results obtained for an environment using only Cloud.

Application amount	Access time average (ms)	Standard deviation (ms)
1	523.66	138.04
10	526.42	133.31
100	1891.36	130.64

Table 3: Results obtained for an environment using Cloud and Fog.

Application amount	Access time average (ms)	Standard deviation (ms)
1	16.71	0.79
10	27.26	0.67
100	288.51	5.77

Table 4: Results obtained for an environment using Cloud, Fog, and Blockchain.

Application amount	Access time average (ms)	Standard deviation (ms)
1	33.04	3.24
10	36.67	3.45
100	331.05	5.82

Fog Node. However, a query is made in Ethereum to verify if the module that wants to manipulate a given data has one transaction authorizing it. Table 4 shows the results obtained for this environment.

5.3. Answers to the Research Questions. This section answers the research questions defined in Section 5.1.

5.3.1. RQ1: Does the Use of Fog Computing Improve the Access Time to the Patient's Medical Records? We used the test results shown in Tables 2 and 3 to answer this question. This was done with the objective of evaluating the impacts generated by the use of Fog Computing in the optimization of the time for accessing medical records. Tests revealed that the use of Fog Computing positively favors performance, as can be seen in Figure 16. This improvement already represents a considerable impact on the set of 10 applications. In addition, the difference in average response time begins to gain greater representativeness as the number of applications grows. With the amount of 100 applications, the average time for the environment that used Fog is 6.5 times smaller compared to the environment that uses only Cloud. This fact demonstrates that the use of Fog optimizes performance related to access time to medical records compared to approaches that use only Cloud infrastructures.

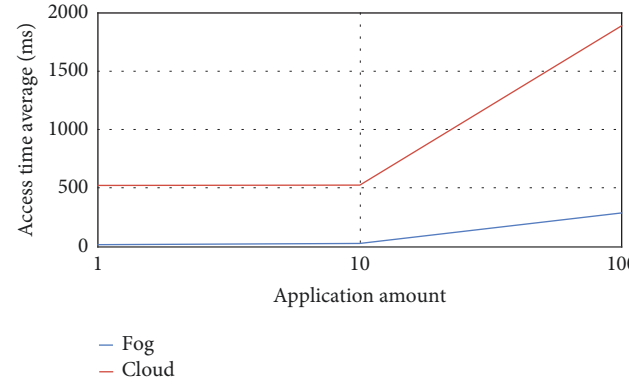


Figure 16: Comparison of average access time using only Cloud versus Fog paradigm.

5.3.2. RQ2: Does the Blockchain-Based Privacy Control Strategy Impair the Access Time to the Patient's Medical Records? We used the test results shown in Tables 3 and 4 to answer

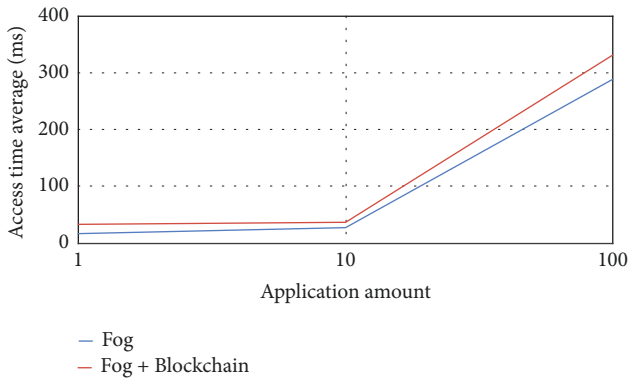


FIGURE 17: Comparison of average access time using Fog versus Fog with Blockchain-based privacy strategy.

this question. This was done in order to measure the performance impacts generated by the addition of Blockchain-based privacy control strategy. Thus, it was tried to verify if the use of Blockchain represents a threat to the performance of the approach proposed in this article. As expected, the graph of the Figure 17 shows that the Blockchain-based strategy adds a small impact to the time of access to medical records. However, for the test with 100 applications, the average time with the use of Blockchain increased only 42.53 ms compared to the environment that did not use this technology, representing an increase of 14.74%. Therefore, it was concluded that the access time to the patient's medical records was not significantly impacted by the privacy control strategy defined in this study.

5.3.3. RQ3: Does the Proposed Approach Allow the Patient to Restrict Third-Party Access to His/Her Medical Records? The approach proposed in this paper offers patients real possession of their medical records, allowing them to control applications that can access their data and subsets of information that they can manipulate. In Figure 15(b), the patient application grants access to Nurse Application only to data collected by E-Health Shield. Thus, the Nurse Application is allowed to access this data, as shown in Figure 14(b). In contrast, Fog Node will deny access to this application if it tries to access another subset of that patient's information. Figure 18 shows an access attempt of the Nurse Application to the patient's blood test data. In this case, Fog returns an error message to Nurse Application.

5.3.4. RQ4: Does the Proposed Approach Allow Different Systems to Exchange Information? With the scenario implemented, it was possible to prove that the proposed approach allows different applications to exchange data. This capability is represented by the schema in the Figure 19, which shows that data collected by E-Health Shield is sent to Raspberry Pi. Therefore, this device sends this information to Fog. This layer provides a REST interface allowing different types of applications to access that data, providing the syntactic interoperability of the proposed solution. Finally, the patient's mobile application and the nurse's web application interpret this information and show it to their users, as shown in Figures 13(c) and 14(b), providing the semantic interoperability.

5.4. Threats to Validity. Three types of threats to validity defined in [48] were evaluated for this case study:

(i) *Construct validity*: the guides defined in [46–48] were followed to avoid threats in the construction of this case study, supporting the planning, conduction, analysis, and reporting of the same. In addition, the planning of the case study was reviewed by four researchers, ensuring its correct execution.

(ii) *External validity*: this paper shows the documentation of the proposed approach and the implementation of the scenario used in the case study. This information can be used to assess the approach in other health scenarios. On the other hand, misunderstanding of the projected architecture or the Fog Computing paradigm or Blockchain technology may compromise the nonfunctional requirements of the proposed approach.

(iii) *Reliability*: this study uses a protocol based on the guides defined by [46–48] to guarantee the reliability of the results. Moreover, quantitative data were collected in several samples, preventing deviations reflecting only a specific moment. Finally, demonstrations of the functioning of the scenario were shown, helping in the evaluation process.

6. Conclusions

This paper describes a software approach designed to enable the management of medical records. It is based on the *Fog Computing* paradigm, providing the availability and performance characteristics by storing the information closer to the applications and devices. Also, it provides privacy through the use of a *Blockchain* infrastructure. The approach also addresses the interoperability requirement by using the REST pattern in the *Communication* module, described in Section 4.2.2. Thus, it intends to facilitate the integration of applications using the different data formats identified in Figure 1.

The study also showed a detailed description of the proposed software architecture, which can be used by analysts, architects, and programmers to build an approach to managing medical records. The solution enables the patient to be the manager of his/her information, controlling which systems can manipulate the data stored in his/her medical record. Also, the patient has greater flexibility to use his/her medical record, since he/she can share the data stored with any other system by using the defined behaviors.

The use of Blockchain in the proposed architecture allows Fog Nodes to carry out the authorization process in a distributed way, eliminating the single point of failure of the traditional authentication model with the Cloud Computing paradigm and giving autonomy so that each Fog Node can function independently and self-contained.

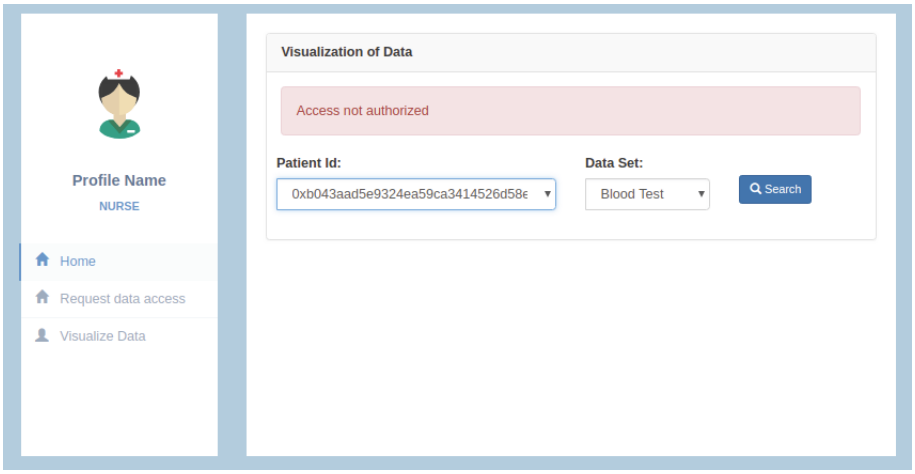


Figure 18: Nurse application attempting to view a subset of data that it has not authorization (Female Medical Nurse Flat Icon Vector.svg from Wikimedia Commons. Author: Videoplasty.com. License: Creative Commons Attribution-ShareAlike 4.0).

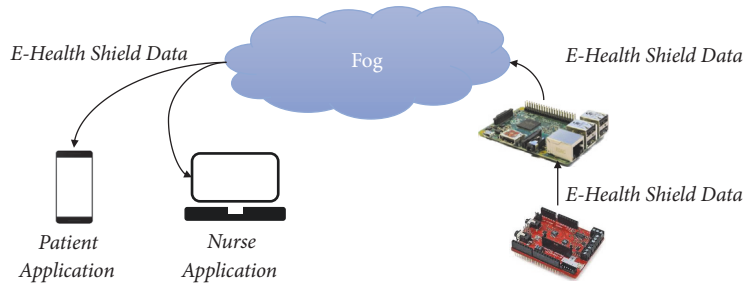


Figure 19: Interoperability on the scenario.

Moreover, the proposed approach was evaluated in a case study, which used a home-centered healthcare scenario to evaluate three nonfunctional requirements of the architecture. The results showed that the architecture was able to provide performance, privacy, and interoperability in the analyzed scenario.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank the Department of Informatics and Applied Mathematics (DIMAp), the Coordination of Improvement of Higher Level Personnel (CAPES), and the National Institute of Software Engineering (INES) for the assistance and support provided in all stages of this work. This work was funded by the Coordination of Improvement of Higher Level Personnel (CAPES) and the National Institute of Software Engineering (INES) [CNPq Grant 465614/2014-0 and FACEPE Grants APQ-0399-1.03/17 and PRONEX APQ/0388-1.03/14].

References

[1] C. A. Silva and G. S. de Aquino Junior, "Fog computing in healthcare: a review," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC '18)*, pp. 01126–01131, 2018.

[2] S. Das, M. Ballav, and S. Karfa, "Application of IoT in detecting health risks due to flickering artificial lights," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI '15*, pp. 2331–2334, IEEE, Kochi, India, 2015.

[3] L. M. R. Tarouco, L. M. Bertholdo, L. Z. Granville et al., "Internet of things in healthcare: interoperatibility and security issues," in *Proceedings of the IEEE International Conference on Communications, ICC '12*, pp. 6121–6125, IEEE, Ottawa, ON, Canada, 2012.

[4] X. Fafoutis, E. Tsimbalo, E. Mellios et al., "A residential maintenance-free long-term activity monitoring system for healthcare applications," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, p. 31, 2016.

[5] World Health Organization, *Global Status Report on Noncommunicable Diseases 2014*, World Health Organization, Geneva, Switzerland, 2014.

[6] World Health Organization, *New Perspectives on Global Health Spending for Universal Health Coverage*, 2017, http://apps.who.int/iris/bitstream/handle/10665/259632/WHO-HIS-HGF-HFWorkingPaper-17.10-eng.pdf.

[7] Deloitte, *2018 Global Health Care Outlook: The Evolution of Smart Health Care*, 2018, https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Life-Sciences-Health-Care/gx-lshc-hc-outlook-2018.pdf.

[8] J. Kharel, H. T. Reda, and S. Y. Shin, "An architecture for smart health monitoring system based on fog computing," *Journal of Communications*, vol. 12, no. 4, pp. 228–233, 2017.

[9] F. A. Kraemer, A. E. Braten, N. Tamkittikhun, and D. Palma, "Fog computing in healthcare-a review and discussion," *IEEE Access*, vol. 5, pp. 9206–9222, 2017.

[10] O. Bibani, C. Mouradian, S. Yangui et al., "A demo of iot healthcare application provisioning in hybrid cloud/fog environment," in *Proceedings of the 8th IEEE International Conference on Cloud Computing Technology and Science, CloudCom '16*, pp. 472–475, IEE, Luxembourg, 2016.

[11] S. Chakraborty, S. Bhowmick, P. Talaga, and D. P. Agrawal, "Fog networks in healthcare application," in *Proceedings of the 13th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS '16*, pp. 386-387, IEEE, Brasilia, Brazil, 2016.

[12] M. Maksimović, "Improving computing issues in internet of things driven e-health systems," in *Proceedings of the International Conference for Young Researchers in Informatics, Mathematics and Engineering '17*, CEUR Workshop Proceedings, pp. 14–17, Kaunas, Lithuania, 2017.

[13] S. Ali and M. Ghazal, "Real-time heart attack mobile detection service (rhamds): an iot use case for software defined networks," in *Proceedings of the 30th IEEE Canadian Conference on Electrical and Computer Engineering, CCECE '17*, pp. 1–6, IEEE, Windsor, ON, Canada, 2017.

[14] 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 *Proceedings of the 1st IEEE International Conference on Fog and Edge Computing, ICFEC '17*, pp. 41–50, IEEE, Madrid, Spain, 2017.

[15] B. Farahani, F. Firouzi, V. Chang et al., "Towards fog-driven iot ehealth: promises and challenges of iot in medicine and healthcare," *Future Generation Computer Systems*, vol. 78, pp. 659–676, 2018.

[16] A. M. Rahmani, T. N. Gia, B. Negash et al., "Exploiting smart e-health gateways at the edge of healthcare internet-of-things: a fog computing approach," *Future Generation Computer Systems*, vol. 78, pp. 641–658, 2018.

[17] C. Dupont, R. Giaffreda, and L. Capra, "Edge computing in iot context: horizontal and vertical linux container migration," in *Proceedings of the Global Internet of Things Summit, GIoTS '17*, pp. 1–4, IEEE, Geneva, Switzerland, 2017.

[18] D. Singh, G. Tripathi, A. M. Alberti, and A. Jara, "Semantic edge computing and iot architecture for military health services in battlefield," in *Proceedings of the 14th IEEE Annual Consumer Communications and Networking Conference, CCNC '17*, pp. 185–190, IEEE, Nevada, Nev, USA, 2017.

[19] I. Azimi, A. Anzanpour, A. M. Rahmani et al., "Medical warning system based on internet of things using fog computing," in *Proceedings of the International Workshop on Big Data and Information Security, IWBIS '16*, pp. 19–24, IEEE, Jakarta, Indonesia, 2016.

[20] A. Limaye and T. Adegbija, "A workload characterization for the internet of medical things (iomt)," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI, ISVLSI '17*, pp. 302–307, IEEE, Bochum, Germany, 2017.

[21] S. Distefano, D. Bruneo, F. Longo et al., "Personalized health tracking with edge computing technologies," *BioNanoScience*, vol. 7, no. 2, pp. 439–441, 2017.

[22] X. Masip-Bruin, E. Marin-Tordera, A. Gómez et al., "Will it be cloud or will it be fog? f2c, a novel flagship computing paradigm for highly demanding services," in *Proceedings of the Future Technologies Conference, FTC '16*, pp. 1129–1136, IEEE, California, Calif, USA, 2016.

[23] X. Masip-Bruin, E. Marin-Tordera, A. Alonso, and J. Garcia, "Fog-to-cloud computing (F2C): the key technology enabler for dependable e-health services deployment," in *Proceedings of the Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net '16)*, pp. 1–5, IEEE, Vilanova i la Geltru, Spain, 2016.

[24] C. S. Nandyala and H.-K. Kim, "From cloud to fog and IoT-based real-time U-healthcare monitoring for smart homes and hospitals," *International Journal of Smart Home*, vol. 10, no. 2, pp. 187–196, 2016.

[25] Y. Shi, G. Ding, H. Wang, H. E. Roman, and S. Lu, "The fog computing service for healthcare," in *Proceedings of the 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare, Ubi-HealthTech '15*, pp. 1–5, IEEE, Beijing, China, 2015.

[26] M. Aazam and E.-N. Huh, "E-HAMC: leveraging fog computing for emergency alert service," in *Proceedings of the 13th IEEE International Conference on Pervasive Computing and Communication, PerCom Workshops '15*, pp. 518–523, IEEE, Missouri, Mo, USA, 2015.

[27] X. Chen and L. Wang, "Exploring fog computing-based adaptive vehicular data scheduling policies through a compositional formal method - PEPA," *IEEE Communications Letters*, vol. 21, no. 4, pp. 745–748, 2017.

[28] A. Alsaffar, P. Hung, and E.-N. Huh, "An architecture of thin client-edge computing collaboration for data distribution and resource allocation in cloud," *International Arab Journal of Information Technolog*, vol. 14, no. 6, pp. 842–850, 2017.

[29] A. A. Alsaffar, H. P. Pham, C.-S. Hong, E.-N. Huh, and M. Aazam, "An architecture of iot service delegation and resource allocation based on collaboration between fog and cloud computing," *Mobile Information Systems*, vol. 2016, Article ID 6123234, 15 pages, 2016.

[30] M. A. Khan and K. Salah, "IoT security: review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[31] M. Swan, *Blockchain: Blueprint for a New Economy*, O'Reilly Media, Inc., 2015.

[32] S. Nakamoto, *Bitcoin: A Peer-To-Peer Electronic Cash System*, 2008, https://bitcoin.org/bitcoin.pdf.

[33] J. Kishigami, S. Fujimura, H. Watanabe, A. Nakadaira, and A. Akutsu, "The blockchain-based digital content distribution system," in *Proceedings of the 5th IEEE International Conference on Big Data and Cloud Computing, BDCloud '15*, pp. 187–190, IEEE, 2015.

[34] M. Samaniego and R. Deters, "Using blockchain to push software-defined iot components onto edge hosts," in *Proceedings of the International Conference on Big Data and Advanced Wireless Technologies, BDAW '16*, p. 58, ACM, 2016.

[35] X. Xu, I. Weber, M. Staples et al., "A taxonomy of blockchain-based systems for architecture design," in *Proceedings of the IEEE International Conference on Software Architecture (ICSA '17)*, pp. 243–252, IEEE, 2017.

[36] H. F. Atlam, A. Alenezi, M. O. Alassafi, and G. B. Wills, "Blockchain with internet of things: benefits, challenges, and future directions," *International Journal of Intelligent Systems and Applications*, vol. 10, no. 6, pp. 40–48, 2018.

[37] M. Samaniego and R. Deters, "Internet of smart things - iost: using blockchain and clips to make things autonomous," in *Proceedings of the 1st IEEE International Conference on Cognitive Computing, ICCC '17*, pp. 9–16, IEEE, 2017.

[38] M. Conoscenti, A. Vetro, and J. C. De Martin, "Blockchain for the internet of things: a systematic literature review," in *Proceedings of the 13th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA '16*, pp. 1–6, IEEE, 2016.

[39] R. Böhme, N. Christin, B. Edelman, and T. Moore, "Bitcoin: economics, technology, and governance," *Journal of Economic Perspectives (JEP)*, vol. 29, no. 2, pp. 213–238, 2015.

[40] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-ng: a scalable blockchain protocol," in *NSDI*, pp. 45–59, 2016.

[41] M. Mettler, "Blockchain technology in healthcare: the revolution starts here," in *Proceedings of the 18th IEEE International Conference on e-Health Networking, Applications and Services, Healthcom '16*, pp. 1–3, IEEE, 2016.

[42] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, United States: Addison-Wesley Professional, 3rd edition, 2012.

[43] D. Solove, *Understanding Privacy*, Harvard University Press, 2008.

[44] P. Clements, F. Bachmann, and L. Bass, *Documenting Software Architectures: Views and Beyond*, United States: Addison-Wesley, 2nd edition, 2010.

[45] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, McGraw-Hill Higher Education, 7th edition, 2011.

[46] B. Kitchenham, L. Pickard, and S. L. Pfleeger, "Case studies for method and tool evaluation," *IEEE Software*, vol. 12, no. 4, pp. 52–62, 1995.

[47] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.

[48] R. K. Yin, *Case Study Research Design and Methods*, vol. 5 of *Applied social research methods series*, 3rd edition, 2003.