A Decentralized Blockchain-based Architecture for a Secure Cloud-Enabled IoT

Mbarek Marwan[1,*], Abdelkarim Ait Temghart[2], Fatima Sifou[3] and Feda AlShahwan[4]

[1]*LTI Laboratory, ENSA, Chouaïb Doukkali University, Morocco*
[2]*TIAD Laboratory, FST, Sultan Moulay Slimane University, Beni Mellal, Morocco*
[3]*LRIT Laboratory, Faculty of Science, Rabat, Morocco*
[4]*College of Technological Studies, Shuwaikh, Kuwait*
E-mail: marwan.mbarek@gmail.com
[*]*Corresponding author*

Received 03 June 2020; Accepted 04 September 2020;
Publication 27 November 2020

Abstract

The integration of cloud computing and Internet of Things (IoT) offers a promising, rich platform for data collection and analysis in smart healthcare. In such a model, IoT devices collect data about patient health status through multiple intelligent sensors, whereas cloud offers scalable resources to quickly meet workload demands. Despite these remarkable improvements, the current architectures do not sufficiently address the security needs for patient medical records. In this perspective, and bearing in mind the specific characteristics of each technology, we propose a distributed security mechanism in a way that fits with IoT and cloud constraints. Our contribution to secure cloud-enabled IoT is twofold. First, we rely on OM-AM (Objective, Model, Architecture and Mechanism) for modeling and analysing the security and privacy requirements of smart healthcare. Second, we use blockchain architecture along with Attribute-Based Access Control (ABAC) model as a decentralized flexible system to support access control decisions. In particular, we rely on XACML (eXtensible Access Control Markup Language) to easily build and implement robust policies required for maintaining a secure IoT-based environment. The novelty of the proposed framework lies at smartly leveraging the recent technologies to keep health information confidential. In fact, putting blockchain and IoT together would undoubtedly create a totally new solution for remote patient monitoring. The simulation results show that the proposal is an efficient way of implementing ubiquitous and cognitive tools for smart healthcare systems.

Keywords: Cloud-enabled IoT, OM-AM model, blockchain, XACML, ABAC, access control.

1 Introduction

Fundamentally, remote sensing medical devices generate vast amounts of data with different formats in their daily operations. Considering the devices' resource scarcity, they all face continuing challenges of energy storage capacities, memory resource and processors' computational power. In this respect, we suggest a novel IT paradigm, called also cloud-enabled Large-Scale Sensor Networks (LSNs), in which cloud computing and IoT represent two complementary technologies, and there would be a big advantage if such two components are integrated and linked together. Certainly, adopting an hybrid architecture can improve efficiency and reliability while providing the right abstraction level [1]. In this respect, the raw medical data, generated by one or more smart IoT devices, are directly sent to the available public cloud for distributed data processing as illustrated in Figure 1.

Subsequently, cloud providers transform sensor data into a real-time clinical feedback. Thus, the generated clinical information is finally delivered to doctors for enhancing decision making [2].

Although cloud facilitates data management and offers scalable and affordable computational resources, the integration of cloud with IoT faces serious challenges related to security, availability and latency [3]. In general, availability problem occurs mainly because of lack of standards, load balancing management, reliability, existence of different heterogeneous technologies, data synchronization, portability and interoperability [4, 5]. More specifically, security concern is the most prominent factor limiting the ubiquitous adaption of IoT technology in the healthcare domain [6]. Current security and privacy measures are often incapable of dealing with this complex environment. In the literature, several various successful security guidelines and procedures have been proposed for distributed systems. Among these models are Multics Rings, Layered Abstractions, Network Protocol Stacks, Napolean Layers, RoFi Layers, Waterfall Model and OM-AM (Objective,

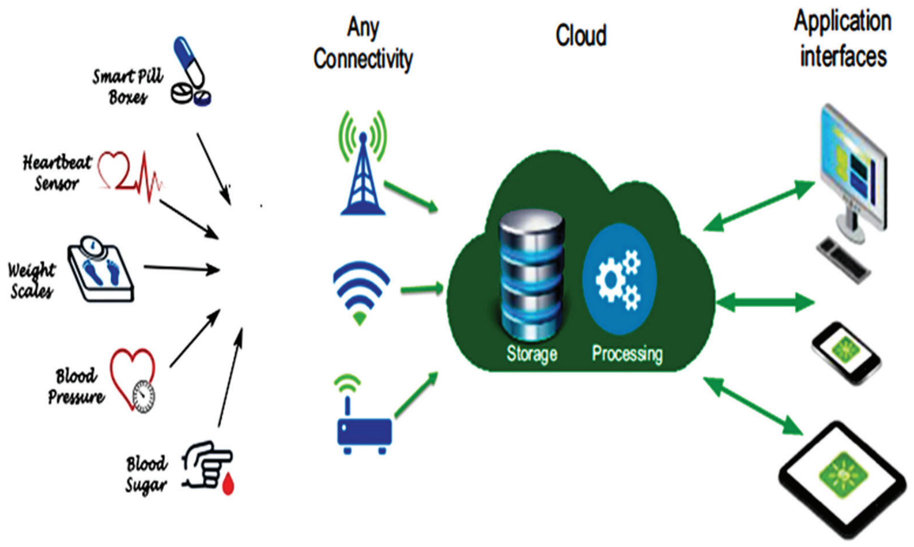


Figure 1 Integration of cloud and IoT in smart healthcare.

Model, Architecture and Mechanism) [7]. For large-scale IoT applications, we suggest the OM-AM model as a guideline for analyzing security risks and privacy requirements of IT applications. In fact, it is a simple and efficient model with only two parameters, namely authority and trust. Usually, the authority to carry out some actions is coupled with the trust that the privilege will be performed correctly. For this reason, authority should be administered and enforced, while the trust requires to be monitored in the context of performing verification and validation. Concretely, the objective and model (OM) layers define the security goals, strategic objectives of the proposed model, solutions to acheive the goal easily. At the same time, the architecture and mechanism (AM) layers must explain in detail the procedures and techniques used to meet the requirements of distributed systems. Based on this model, we suggest an efficient access control that allows the utilization of smart devices while simultaneously keeping data safe when using cloud-enabled IoT. To this aim, we use both blockchain technology and Attribute-Based Access Control (ABAC) model to build an efficient and flexible access control system. As revolutionary as they are already, blockchain and ABAC model have the potential to be much more robust and powerful when combined together. Both can augment each other's capabilities as well as improve the level of the security and trust in a distributed system. Given the fact that blockchain is essentially a distributed ledger to stores confidential data, combined with XACML-based policy, there is no doubt that medical information colleted by smart devices can be transferred and processed securely on remote servers.

Currently, there are already several blockchain-based solutions to manage medical data. For instance, Medrec [8] is a decentralized system to support electronic medical records (EMRs) by using blockchain technology. Technically, the proposed framework provides services such as authentication, confidentiality and data sharing. In the same line, MeDShare [9] relies on smart contracts and an access control mechanism to create a more secure environment for improving collaborative care and clinical data sharing. The work in [10] proposes a blockchain-based framework for a secure and privacy-preserving data management in e-health systems. Concretely, the consortium blockchain creates secure index for each medical data. In this case, public key encrypted with keyword search (PEKS) scheme is used to protect medical data against unauthorized access. Despite the fact that many blockchain-based systems are available, the majority of them do not take into consideration the constraints of IoT devices. The best way to ensure a good balance between security and performance is to store the security policy across different nodes attached to IoT devices. In this paper, leveraging blockchain technology, we suggest a novel decentralized framework composed of cloud and IoT system. This will undoubtedly help IoT devices process data in remote servers securely.

The remainder of this paper is organized as follows. In Section 2, we explore and analyze existing access control models to correctly select the most appropriate one. Section 3 illustrates the proposed model to meet the security requirements in the cloud-enabled IoT. Section 4 provides the main parameters to configure and customize different components of the proposed access control. Finally, we end this paper in Section 5 by remarks and future work.

2 Existing Access Control Models

There is a wide variety of access control models with operational tasks and procedures to define permissions and authorizations of access requests to remote resources. The most commonly used methods to prevent unauthorized access to critical environments are based on different models, namely Mandatory access control (MAC), Discretionary Access Control (DAC), Role-Based Access Control (RBAC) and ABAC [11]. Table 1 summarizes the key factors

Table 1 Advantages and disadvantages of existing access control systems

Models	Pros	Cons
DAC	– It enables faster development and deployment. – It offers flexible security control mechanisms.	– It is limited in its scalability. – There is a problem with roles explosion. – There is always the possibility of errors in the system configuration.
MAC	– It was considered the most secure mechanism for centralized systems. – It is a relatively easy-to-use security countermeasure.	– It is not flexible enough to support distributed systems. – It integrates insufficient security mechanisms to protect data in remote platforms.
RBAC	– It is a very useful tool for satisfying the requirements of large systems. – It is used to effectively manage authorizations based on user's requests and security policy.	– It requires additional tasks for maintenance of existing users' roles. – It suffers from role explosion. – It cannot support the constraints linked to real-time computing.
ABAC	– It offers dynamic security mechanisms. – It provides a contextual framework to enhance security policies development. – It has the ability to offer scalable fine grained access.	– There is a computational overhead associated with complex security policies.

in selecting the most appropriate access control model for the public cloud environments [12].

In view of the above findings, it is necessary to identify the key indicators that will help us to identify the weaknesses and strengths of each access control scheme in order to select the appropriate model for the cloud computing and IoT environment. In this study, we rely on the following parameters to compare existing security models; among these, dynamicity, flexibility, reliability, easy-to-use, running time, easy in administration, support scaling rules and fine-grained access [12].

In general, each access control model is tailored to meet the needs of security and performance requirements. In this context, ABAC and RBAC models are absolutely appropriate to support fine-grained access control to fulfill a completely secure data sharing scheme. In the same line, MAC and DAC are lightweight solutions to significantly reduce the computational complexity of access decisions. However, MAC and DAC are absolutely inappropriate for dynamic and flexible environment such as cloud and IoT.

For a comprehensive comparative analysis, we first use a seven-level likert scale to describe the appropriateness level for the examined access control models, when they are used for specific purposes.

1. Absolutely inappropriate
2. Inappropriate
3. Slightly inappropriate
4. Neutral
5. Slightly appropriate
6. Appropriate
7. Absolutely appropriate

Next, we examine the level of appropriateness of those models based on the selected criteria, as shown in Table 2.

In light of this fact, the utilisation of ABAC is considered to be the most secure, flexible and scalable tool since it meets the required properties of distributed systems. This is a result of the fact that ABAC offers a hierarchical structure to facilitate access control in a dynamic environment. Although the existing ABAC schemes have made significant achievements in addressing security, in some aspects, further improvements are needed to meet the requirements of cloud-enabled IoT, particularly those linked to flexibility, efficiency and adaptability in the context of distribution systems.

Table 2 A comparative study on access control models

	MAC	DAC	RBAC	ABAC
Dynamicity	1	1	5	6
Flexibility	1	2	4	7
Reliability	4	6	6	6
Easy-to-use	6	6	3	5
Policy management	6	5	4	6
Scaling rules	2	3	5	6
Computational complexity	4	5	5	4
Fine-grained data sharing	2	4	6	7

In this respect, we propose a novel distributed access control based on the blockchain and ABAC model to support the integration of cloud computing with IoT in a secure manner.

3 Fundamentals of the Proposed Solution

The purpose of the new model is to provide a flexible access control that allows distributed IoT devices to safely outsource their data. In this respect, we rely on the OM-AM model to design the appropriate access control for cloud-enabled IoT. More importantly, we use both blockchain and ABAC to achieve this goal with respect to technical constraints.

3.1 Building Blocks

The OM-AM model [13] contains four layers in the following order: objective, model, architecture and mechanism layers. To design the most appropriate security mechanism, the OM-AM model separates the proposed solution into various levels, and then defines each layer more clearly and completely. In a top-down approach, we begin with a description of an overview of the suggested system and formulate its specifications. Thus, we start from the high-level to present an overview of the system and then we describe enforcement mechanisms and implementation at the low-level layers [14, 15]. Figure 2 depicts the layers of the OM-AM model. From the top they are, Objective layer, Model layer, Architecture layer and Mechanism layer.

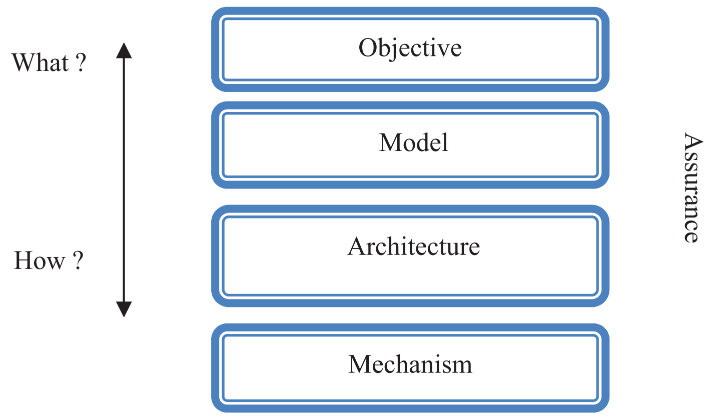


Figure 2 The OM-AM model [13].

- Objective layer: the primary purpose of this layer is to describe various aspects of the security policy that will be used to meet privacy requirements. Specifically, this layer consists in defining rules of high-level access control policies in the context of cloud-enabled IoT.
- Model layer: the aim is to formalize the specified level of assurance, which is defined in the objective layer. Hence, it builds a bridge between the high-level security policies and low-level mechanisms. In the OM-AM model, it is recommended to rely on existing approaches and techniques, rather than reinventing the wheel every time. In this regard, we select the appropriate security policies to derive optimal sets of security mechanisms, based upon a comprehensive security threats survey. To this objective, we use customization approach to reshap a general model to yield a specific solution. Thus, we define means and measures that should be applied to protect remote resources. In brief, this layer presents a general formalism for representing the proposed security policy in a clear way.
- Architecture layer: the goal here is to efficiently design and build a robust IT security infrastructure, representing the different components of the proposed framework. In this context, we explore and analyze various possible architectural solutions for effectively implementing the appropriate security mechanism. For instance, fine-grained access control schemes are commonly used in the cloud and IoT systems to protect confidential data.
- Mechanism layer: it determines a variety of mechanisms and protocols that basically support these architectures to achieve a higher level of maturity regarding information security. Moreover, it provides a clear, simple overview of how clients' requests are evaluated against the implemented security policies.

In the same vein, we use ABAC model to minimize the risk of unauthorized access to cloud ressources. This framework is composed mainly of a set of components namely Policy Enforcement Point (PEP), Policy Information Point (PIP), Policy Administration Point (PAP) and Policy Decision Point (PDP) layer [16]. Figure 3 gives a basic overview of the suggested access control and the authorization decision process.

In this case, each component has been developed to capture the security requirements precisely. In this model, the administrator uses PAD interface to directly define and store security policies in the Security Policies Repositry (SPR). Similarity, the PEP entity intercepts the original requests and converts

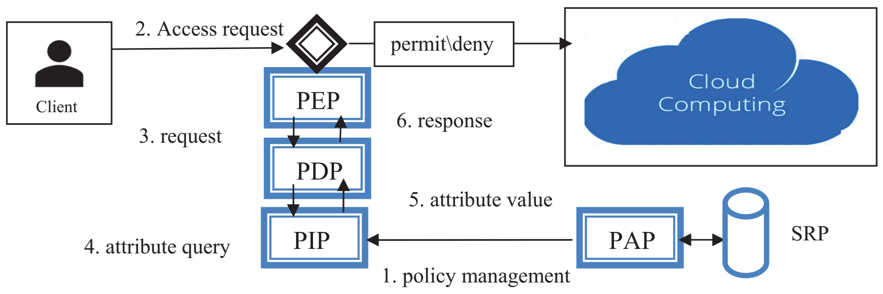


Figure 3 XACML decision architecture [17].

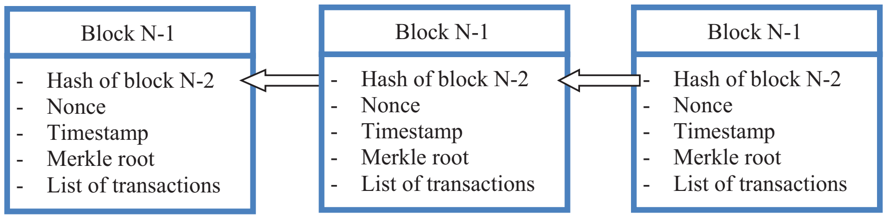


Figure 4 Simplified data structure in blockchain technology.

them into XACML format. In this case, these requests are evaluated by the PDP module to support effective decision making according to the implemented security policies. To this aim, the PIP entity is used to define a set of attributes related to objects and subjects; this, subsequently, helps the system evaluate each access request.

As outlined above, blockchain framework is considered the core element of the proposed system. It is essentially an asset database that can be easily shared across a network that is composed of multiple IoT devices. Moreover, it performs some extra tasks for managing data confidentiality, particularly those related to signature verification, a consensus mechanism and redundancy. This distributed ledger stores its transactional data into a block and all blocks are linked together to form a chain of blocks [18]. In this case, every block on the blockchain contains a link to the hash of the previous block, as presented in Figure 4.

3.2 The Proposed Models

The primary purpose of this new model is to design a light-weight access control mechanism that protects confidential information so as to comply with privacy legislations. To this aim, the OM-AM reference is adopted as an effective approach to design, implement and maintain a security policy. This case study deals with a cloud-based platform for distributed IoT containing several autonomous devices. It is based on the requirements of healthcare applications in terms of security and privacy. More specifically, the proposed framework provides a dynamic, efficient access control for the cloud-enabled IoT (AC-CIoT) using each of the four layers of OM-AM model.

3.2.1 AC-CIoT objective

The proposed system is composed of several smart sensor nodes, each of which is connected to cloud computing. In this context, it is necessary to have an identifier that uniquely identifies each device. Besides, access control is used to protect the secret data of IoT devices. This could be achieved through multiple roles possessed by the node that tries to gain access to cloud resources. Accordingly, the major challenges faced by all security policies are with the management and supervision of the relation between user-role and role-permission. It is assumed that the AC-CioT system determines what permissions are assigned to each role, and allows the revocation of these permissions at any time. Consequently, we need to design and implement the appropriate policy to cop with any eventual security problems arising in the IoT platform.

3.2.2 AC-CIoT model

Before we consider how to do the objectives stated previously, plans and actions based on clear goals are suggested and evaluated. Therefore, countermeasures are absolutely necessary to prevent the disclosure of confidential data. The access control is the fundamental component of this proposed security measures. The literature review shows that there exist several schemes with multiple access techniques, including DAC, MAC, RBAC and ABAC. However, these classical methods do not support flexible and easy-to-use rights delegation in the context of IoT systems. In this respect, we suggest the utilization of ABAC along with blockchain. Essentially, the ABAC-based access control and blockchain technology provides the appropriate security mechanisms. In fact, this model takes into consideration the requirements of distributed systems and IoT applications. On the one hand, ABAC is responsible for the evaluation of all required parameters of access decisions. These parameters refer usually to attributes of the subject and object, environment and conditions. In this case, the predefined policies are explicitly designed to allow or deny access to specific resources in the IoT platform.

More importantly, ABAC model is an efficient, simple way to express and formulate security policy through the Boolean functions [19]. This model ensures flexibility because it does not require the definition of the previous relationship between subjects and objects [20]. Hence, it is suitable to evaluate access requests efficiently for a decentralized and dynamic environnement like IoT platform. On the one hand, blockchain is used as a public repository of data and also as a policy retrieval point. In this case, this technology provides a secure, available, decentralized database to implement security policies [21]. In other words, all access control policies with pair (resource, requester) are saved as transactions in this distributed platform [22]. Moreover, blockchain provides logging databases to ensure auditing functions and integrity checking mechanism. Concretely, blockchain is used mainly as an authorization manager point (AMP) to access a particular resource that is normally identified by its public address. To this end, we rely on a digital signature, called authorization token, to determine the access right created by the sender of a transaction to its receiver. Usually, resource owners and requesters are identified with their addresses, interacting between them via transactions [23]. In this context, the address is a secure public identifier shared in the network. Basically, the address is used for determining the source and destination of a particular transaction. Technically, the major existing models are built upon a cryptographic system with a pair of keys, namely public key and private key. In general, the private key servers for authentication purposes while the public key is used to helps ascertain the identity of the user or resource so that every node is visible in the network. Formally, we use Pay-to-Public-Key-Hash (P2PKH) as a standard format to represent a public key in a better readable way as in the case of any checksum method [24]. Alternatively, we can use Pay-to-Script-Hash (PSH) to generate an address by using the hash of a script instead of the hash of the public key.

3.2.3 AC-CIoT architecture

The proposed architecture provides a generic platform that uses blockchain platform to safeguard security policies linked to available public resources. At the same time, we rely on the XACML language to express access control rules in the security policy. Figure 5 illustrates the skeleton architecture of the suggested framework.

The building blocks of our framework are as follows:

- Users: it refers to resource's owner or resource requester and basically possesses two keys: public and private. The former are typically saved in the public blockchain. Moreover, the requested resources for each

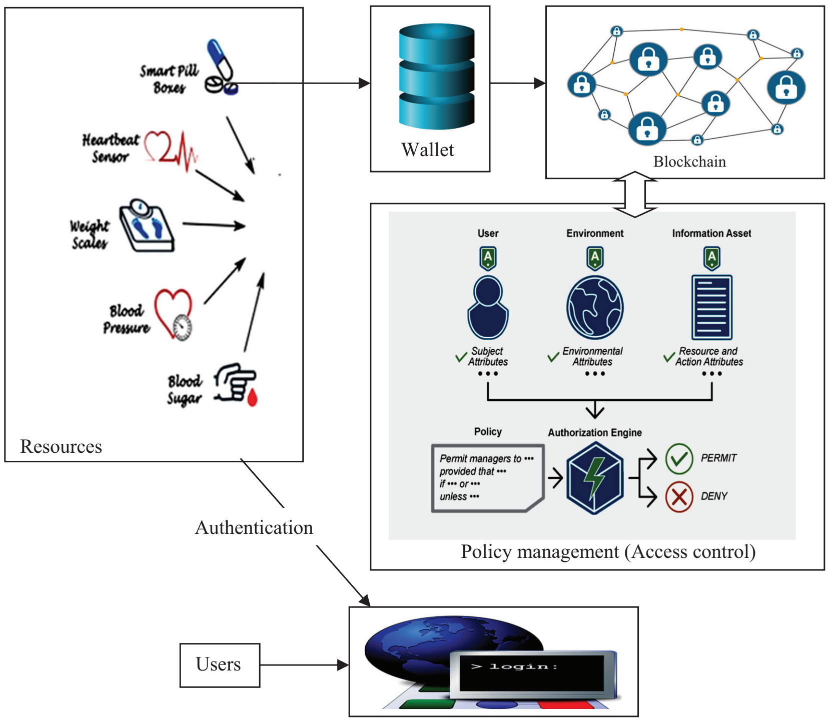


Figure 5 General architecture of blockchain-based access control.

client are saved in a secure database. In this case, the owner who has a legitimate right of access critical resource in the cloud-enabled IoT can create a specific policy on every resource. Subsequently, the URL of this policy is stored in the blockchain database. Accordingly, requester needs to scan the database that is deployed in the public blockchain to get the correct URL of each resource's policy. This approach aims at reducing the number of transactions required to gain access to a specific resource.

- Resources: they refer to things that are useful in detecting or monitoring medical problems. These smart devices measure clinical vital statistics, including heart rate, body temperature, breathing rate, etc. For security purposes, each IoT device has its own unique access control policy, and authorized groups or users. Such security policy is typically developed and implemented by the resource's owner.
- Access control: we rely on ABAC model [25, 26] to prevent unauthorized access to Internet-of-Things of medical devices by using policies.

To this aim, we rely on PIP module to get the required attributes used in making an access control decision. In this case, SPR module is a database or filesystem used as repository to store policies that define the rules by which data and resources are protected against unauthorized access. The PAP component provides an interface to easily add and update security policies.

- Transactions: it records any operation and the exchange of information across different network participants in the blockchain platform. In this context, we can make a request to resource's owner by using Resource Access Transaction (RAT). In the same line, the owner uses Policy Transactions (PT) to add a policy to the database that already contains local policy associated to each resource and access transaction. This strategy would help share confidential data with other users in the cloud-enabled IoT. For each transaction, we use an identifier (address of a user) and the input (type of transactions) to create a new transaction in the blockchain.
- Blockchain [27]: it refers to the database that allows the storage of transactions and locations of both the resources associated with cloud-enabled IoT and the URL's of each device policy. Thus, the blockchain platform consists of a continuous sequence of blocks. In this case, each block is composed of a given timestamp along with a link to the previous block [28].
- Wallet: it is a trusted node associated to each user and device in the blockchain network, and hence acts as a secure tunnel that performs the transaction between clients and blockchain database [29]. Basically, this nodes contains all keys (private and public keys) required by clients so as to define and register their resources, sign their transactions, and ask for access to remote confidential resources available on cloud-enabled IoT. For simplicity reasons, it could be a web-based interface or mobile applications, through which the owner registers his sensitive resources, defines security policies clearly. In this framework, the wallet is designed to ensure three functions. (i) Create keys automatically and protect them. (ii) Translate the global access control policy to several transactions and then dispatched them immediately into the whole blockchain platform. (iii) Select any transaction to check all previous transactions to confirm that cloud-enabled IoT is protected from unauthorized or malicious access. In this context, keys are used for authentication purposes.

3.2.4 AC-CIoT mechanism

Besides access control, additional security measures are implemented to ensure the integrity, authentication and confidentiality of the cloud-enabled IoT. In this regard, we use SSL/TLS as a cryptographic protocol to ensure data integrity for transmissions made over a TCP/IP network. In this case, SSL protocol establishes and maintains trust between cloud computing and IoT devices before medical data transfer happens. Moreover, digital certificates must be kept safe in order to securely send data to cloud computing.

4 Simulation and Typical Use Case

In this study, we will briefly discuss initial experiments related to a proof-of-concept of our proposal so as to demonstrate its feasibility and utility. As explained already, conceptual framework provides an access control for protecting IoT devices and health data against cyber threats. As our access control is built on XACML-based policies, we leverage the use of blockchain solutions as a repository for these created policies. In this respect, we will explain the principle of each technology, which is the key component of the implementation process.

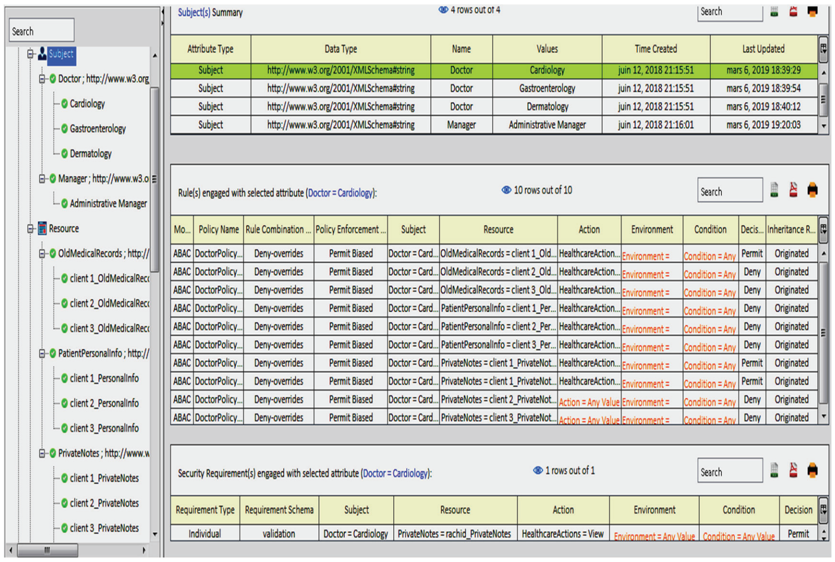
4.1 Blockchain

To successfully implement the blockchain technology, we rely on Bitcoin Core (BC) framework. The latter is a free and open-source tool that contains serveral Bitcoin nodes [30]. Deploying current completed software, especially in version of BC, we usually need 200 GB of disk space. For simplicity purposes, we use BitCoinJ [30] as an open application for Bitcoin protocol, implemented in Java, without need of a local copy of BC. As a testing environment, it is quite effective to implement Bitcoin's test network (testnet) or regression test mode (regtest) for the trusted blockchain platform [31]. In the first scenario, there are no standard for transactions checking, and coins on the testnet have no value and can be obtained for free [32]. In the regtest mode, the interaction with random peers and blocks is unnecessary, and hence all processes are launched on the local host [33, 34]. For the proposed framework, we opt for BitCoinJ tool to deploy the bitcoin protocol in a distributed simulation environment, and regtest for the test mode. In BitCoinJ framework, the Wallet class is one of the most essential components as it safeguards both keys and transactions, which are basically used to assign value to/from those keys. Naturally, there exist several procedures to guatantee the appropriate setting for a successful implementation of blockchain platform. In this context, the most useful commands and scripts are presented briefly in the remainder of this section [35].

- Create a wallet: we use the following commands to crate a new wallet.
  BlockChain chain_1 = new BlockChain (set_params, w_1, ...); // where w_1 is a new wallet
  PeerGroup peerGroup_1 = new PeerGroup (set_params, chain_1);
  peerGroup_1.addWallet (w_1);
- Create a public key (address): for an elliptic curve public key (EK) and a private key (PK), we use the following command to generate the public address:
  public static EK *fromPrivate* (BigInteger *PK*)
  - Create a new address in the regtest mode: to build applications within an IoT platform, we should initially create an address by using the following command:
    bitcoin-cli -regtest get_new_address
  - Create a transaction in the regtest mode: fundamentally, we use bitcoin-cli function and Unspent Transaction Output (UTXO) to generate a transaction "Tr1". This is done by using the following command line:
    Tr1: $UTXO1_Tr1

4.2 XACML-based Policy

To successfully build fine-grained access control, we rely on XACML language [36, 37] to define and implement robust security policies. To show the usefulness and benefits of the proposed approach, we consider a typical use case that describes a scenario involving healthcare delivery systems and the exchange of medical data quickly and simply. In this respect, we use Security Policy Tool [38] to easily design XACML policies for healthcare domain [39]. This tool provides an interactive graphical user interface that allows security managers to customize policy's attributes, including subjects, resources and rules. For instance, we define two types of users as subjects. The first one refers to doctors, especially cardiology, gastroenterology and dermatology. The second one is healthcare managers that handle administrative tasks such as supervising daily administrative operations and monitoring expenses. For resources, we define three main categories of medical data for three clients, i.e, health records, personal information and patient note. Figure 6 shows a formal way to define attributes of the proposed security policy.



Subject(s) Summary   4 rows out of 4

Attribute Type	Data Type	Name	Values	Time Created	Last Updated
Subject	http://www.w3.org/2001/XMLSchema#string	Doctor	Cardiology	juin 12, 2018 21:15:51	mars 6, 2019 18:39:29
Subject	http://www.w3.org/2001/XMLSchema#string	Doctor	Gastroenterology	juin 12, 2018 21:15:51	mars 6, 2019 18:39:54
Subject	http://www.w3.org/2001/XMLSchema#string	Doctor	Dermatology	juin 12, 2018 21:15:51	mars 6, 2019 18:40:12
Subject	http://www.w3.org/2001/XMLSchema#string	Manager	Administrative Manager	juin 12, 2018 21:16:01	mars 6, 2019 19:20:03

Rule(s) engaged with selected attribute (Doctor = Cardiology):   10 rows out of 10

Mo...	Policy Name	Rule Combination ...	Policy Enforcement ...	Subject	Resource	Action	Environment	Condition	Decis...	Inheritance R...
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	OldMedicalRecords = client 1_Old...	HealthcareAction...	Environment =	Condition = Any	Permit	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	OldMedicalRecords = client 2_Old...	HealthcareAction...	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	OldMedicalRecords = client 3_Old...	HealthcareAction...	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PatientPersonalInfo = client 1_Per...	HealthcareAction...	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PatientPersonalInfo = client 2_Per...	HealthcareAction...	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PatientPersonalInfo = client 3_Per...	HealthcareAction...	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PrivateNotes = client 1_PrivateNot...	HealthcareAction...	Environment =	Condition = Any	Permit	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PrivateNotes = client 1_PrivateNot...	HealthcareAction...	Environment =	Condition = Any	Permit	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PrivateNotes = client 2_PrivateNot...	Action = Any Value	Environment =	Condition = Any	Deny	Originated
ABAC	DoctorPolicy...	Deny-overrides	Permit Biased	Doctor = Card...	PrivateNotes = client 3_PrivateNot...	Action = Any Value	Environment =	Condition = Any	Deny	Originated

Security Requirement(s) engaged with selected attribute (Doctor = Cardiology):   1 rows out of 1

Requirement Type	Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision
Individual	validation	Doctor = Cardiology	PrivateNotes = rachid_PrivateNotes	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Permit

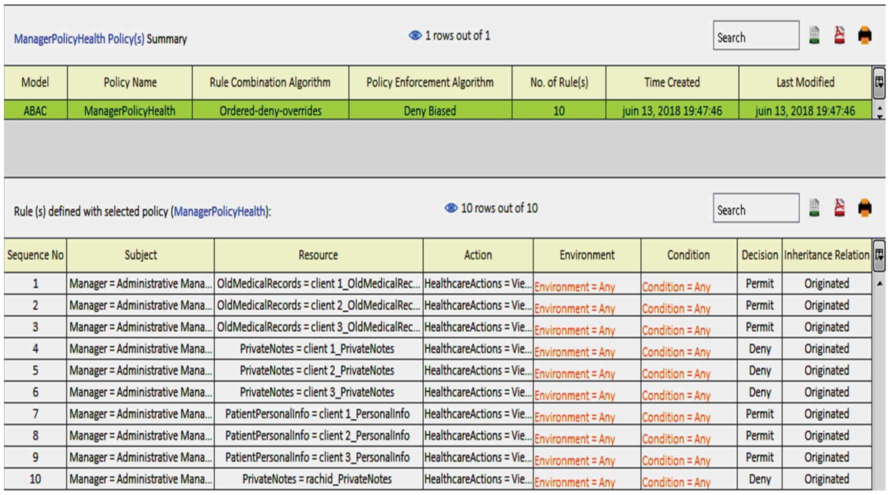
Figure 6 Different types of attributes for subjects and resources.

To enforce medical data protection, we suggest two security policies that allow healthcare organizations to decide which users can view or add health records in cloud-enabled IoT. In this context, we have defined two policies to ensure adequate privacy protection. For managers, they can only view and add administrative data, but they cannot access neither to health records nor patient's notes. Besides, we use ordered-deny-overrides for rule combination algorithm, and deny-biased for policy enforcement algorithm, as illustrated in Figure 7.

To avoid the disclosure of confidential data, we define 10 rules that define a set of possible rules for each resource by adding restrictions on who can access remote data, which medical records a user is allowed to view or add. To this aim, each rule defines the desired effect, either to permit or deny an action, as illustrated in Figure 8.

In the same line, we define the security policy for healthcare professionals (doctors). In this regard, doctors can view medical records and add patient's notes, but they can not view administrative data and personal information. The formalization process is realized via the definition of several rules, as illustrated in Figure 9.

Ideally, we rely on XML-based language to define robust security policies for fine-grained access-control. To this aim, it is necessary to convert the


ManagerPolicyHealth Policy(s) Summary

1 rows out of 1

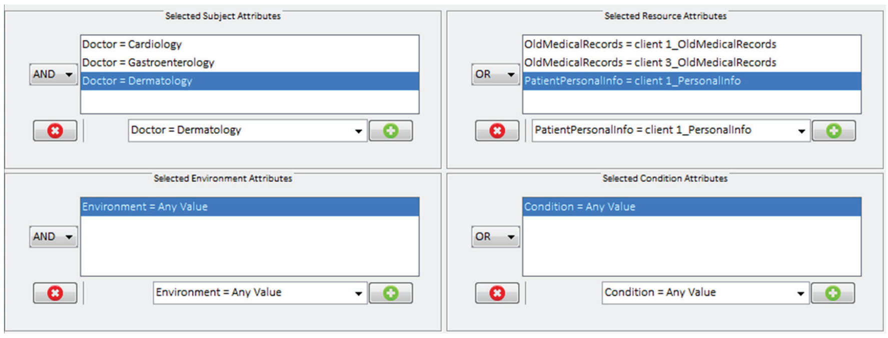
Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	ManagerPolicyHealth	Ordered-deny-overrides	Deny Biased	10	juin 13, 2018 19:47:46	juin 13, 2018 19:47:46

Rule (s) defined with selected policy (ManagerPolicyHealth):

10 rows out of 10

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Manager = Administrative Mana...	OldMedicalRecords = client 1_OldMedicalRec...	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
2	Manager = Administrative Mana...	OldMedicalRecords = client 2_OldMedicalRec...	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
3	Manager = Administrative Mana...	OldMedicalRecords = client 3_OldMedicalRec...	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
4	Manager = Administrative Mana...	PrivateNotes = client 1_PrivateNotes	HealthcareActions = Vie...	Environment = Any	Condition = Any	Deny	Originated
5	Manager = Administrative Mana...	PrivateNotes = client 2_PrivateNotes	HealthcareActions = Vie...	Environment = Any	Condition = Any	Deny	Originated
6	Manager = Administrative Mana...	PrivateNotes = client 3_PrivateNotes	HealthcareActions = Vie...	Environment = Any	Condition = Any	Deny	Originated
7	Manager = Administrative Mana...	PatientPersonalInfo = client 1_PersonalInfo	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
8	Manager = Administrative Mana...	PatientPersonalInfo = client 2_PersonalInfo	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
9	Manager = Administrative Mana...	PatientPersonalInfo = client 3_PersonalInfo	HealthcareActions = Vie...	Environment = Any	Condition = Any	Permit	Originated
10	Manager = Administrative Mana...	PrivateNotes = rachid_PrivateNotes	HealthcareActions = Vie...	Environment = Any	Condition = Any	Deny	Originated

Figure 7 Definition of security policy for managers.


Selected Subject Attributes

AND

Doctor = Cardiology
Doctor = Gastroenterology
Doctor = Dermatology

Doctor = Dermatology

Selected Resource Attributes

OR

OldMedicalRecords = client 1_OldMedicalRecords
OldMedicalRecords = client 3_OldMedicalRecords
PatientPersonalInfo = client 1_PersonalInfo

PatientPersonalInfo = client 1_PersonalInfo

Selected Environment Attributes

AND

Environment = Any Value

Environment = Any Value

Selected Condition Attributes

OR

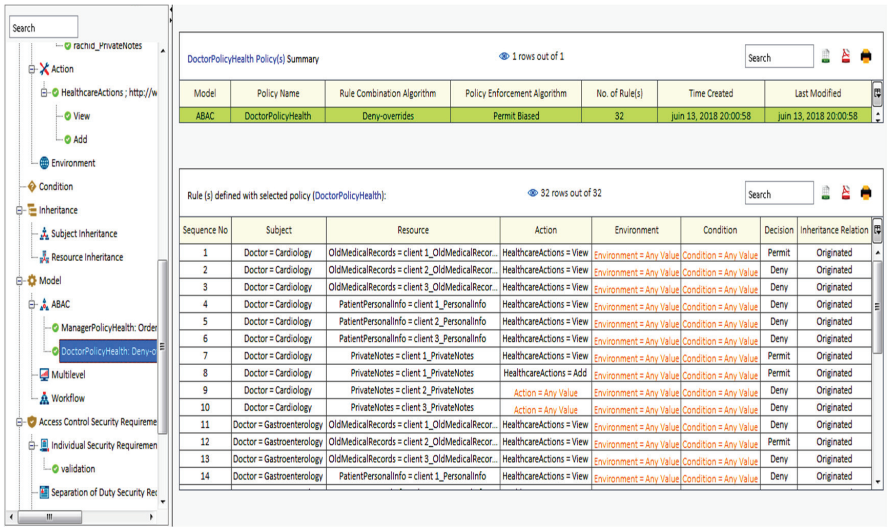
Condition = Any Value

Condition = Any Value

Figure 8 Definition of role associated to managers' security policy.

created policy into different rules written in the XACML syntax. Figure 10 represents a piece of our converted policy that contains 900 rows.

In summary, many solutions have been proposed aiming to prevent unauthorized users from retrieving, using, or altering confidential data. With regards to the ABAC properties, it succeeded in representing correctly the platform composed of a cloud computing infrastructure and IoT devices. A further improvement of this actual solution relies on the adoption of a more distributed architecture, for example blockchain platform as a service. Its benefits are obvious in providing secure storage for global security policies. As far as security and privacy are concerned, we use XACML language


Search
rachid_PrivateNotes
Action
HealthcareActions ; http://w
View
Add
Environment
Condition
Inheritance
Subject Inheritance
Resource Inheritance
Model
ABAC
ManagerPolicyHealth: Order
DoctorPolicyHealth: Deny-o
Multilevel
Workflow
Access Control Security Requireme
Individual Security Requiremen
validation
Separation of Duty Security Rec
Search

DoctorPolicyHealth Policy(s) Summary — 1 rows out of 1

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	DoctorPolicyHealth	Deny-overrides	Permit Biased	32	juin 13, 2018 20:00:58	juin 13, 2018 20:00:58

Rule (s) defined with selected policy (DoctorPolicyHealth): — 32 rows out of 32

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Doctor = Cardiology	OldMedicalRecords = client 1_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Permit	Originated
2	Doctor = Cardiology	OldMedicalRecords = client 2_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
3	Doctor = Cardiology	OldMedicalRecords = client 3_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
4	Doctor = Cardiology	PatientPersonalInfo = client 1_PersonalInfo	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
5	Doctor = Cardiology	PatientPersonalInfo = client 2_PersonalInfo	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
6	Doctor = Cardiology	PatientPersonalInfo = client 3_PersonalInfo	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
7	Doctor = Cardiology	PrivateNotes = client 1_PrivateNotes	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Permit	Originated
8	Doctor = Cardiology	PrivateNotes = client 1_PrivateNotes	HealthcareActions = Add	Environment = Any Value	Condition = Any Value	Permit	Originated
9	Doctor = Cardiology	PrivateNotes = client 2_PrivateNotes	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
10	Doctor = Cardiology	PrivateNotes = client 3_PrivateNotes	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
11	Doctor = Gastroenterology	OldMedicalRecords = client 1_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
12	Doctor = Gastroenterology	OldMedicalRecords = client 2_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Permit	Originated
13	Doctor = Gastroenterology	OldMedicalRecords = client 3_OldMedicalRecor...	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated
14	Doctor = Gastroenterology	PatientPersonalInfo = client 1_PersonalInfo	HealthcareActions = View	Environment = Any Value	Condition = Any Value	Deny	Originated

Figure 9 Definition of security policy for healthcare professionals.

```
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:http://www.w3.org/2001/xmlschema#string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Dermatology</AttributeValue>
            <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0::subjectcategory:accesssubject" AttributeId=
"urn:oasis:names:tc:xacml:1.0:subject:Doctor" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"></
AttributeDesignator>
          </Match>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:http://www.w3.org/2001/xmlschema#string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">client 3_PersonalInfo</AttributeValue>
            <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0::attributecategory:resource" AttributeId=
"urn:oasis:names:tc:xacml:1.0:resource:PatientPersonalInfo" DataType="http://www.w3.org/2001/XMLSchema#string"
MustBePresent="true"></AttributeDesignator>
          </Match>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:http://www.w3.org/2001/xmlschema#string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">View</AttributeValue>
            <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0::attributecategory:action" AttributeId=
"urn:oasis:names:tc:xacml:1.0:action:HealthcareActions" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent=
"true"></AttributeDesignator>
          </Match>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:http://www.w3.org/2001/xmlschema#string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Any Value</AttributeValue>
            <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0::attributecategory:environment" AttributeId=
"urn:oasis:names:tc:xacml:1.0:environment:Environment" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent=
"true"></AttributeDesignator>
          </Match>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:http://www.w3.org/2001/xmlschema#string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Any Value</AttributeValue>
            <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0::attributecategory:condition" AttributeId=
"urn:oasis:names:tc:xacml:1.0:condition:Condition" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"
```

Figure 10 A piece of XACML syntax of the proposed security policy.

to create and implement robust security policies. Therefore, the main goal of the proposed solution is the design and development of a flexible and distributed access control for cloud-enabled IoT architecture. As a result, the proposal is a lightweight system conceived for the distributed and autonomic management, as well as optimizing the process of authorisation decisions. Table 3 provides the benefits and drawbacks of each access control solution in a distributed environment.

Table 3 A brief comparison between existing solutions and the proposed one

	MAC	DAC	RBAC	ABAC	The Proposed Solution
Distributed management	2	2	5	6	7
Scaling rules	2	3	5	6	6
Computational complexity	4	5	5	5	6
Fine-grained data sharing	2	4	6	7	7

5 Conclusion and Future Work

The utilization of cloud-enabled IoT is an emerging technology that empowers resource-scarce devices to get flexible and scalable computaional resources. In smart healthcare, this hybrid architecture still faces several challenges regarding security, performance and availability. In this respect, we suggest an efficient access control based on the OM-AM model to allow or deny access to sensitive data according to the predefined security policy. Concretely, we use XACML for implementing an ABAC-based system. In the same line, we rely on a public blockchain as a distributed ledger containing links to security policies. More precisely, we have presented an effective, decentralized solution in which the security policy could be saved in multiple nodes to achieve high availability and data security. This approach is particularly important for supporting dynamic systems and fine-grained data sharing. As a future work, we intend to extend this study by using a platform with realistic IoT scenarios for early proof-of-concept experiments. To this aim, we plan to use the Raspberry Pi device with Android Things v0.8 as an operating system for implementing our proposed framework.

References

[1] M. Díaz, C. Martín and B. Rubio, 'State-of-the-art, challenges, and open issues in the integration of Internet of Things and cloud computing', Journal of Network and Computer Applications, vol. 67, pp. 99–117, 2016.

[2] Marwan, A. Kartit and H. Ouahmane, 'A cloud based solution for collaborative and secure sharing of medical data', International Journal of Enterprise Information Systems, vol. 14, no. 3, pp. 128–145, 2018.

[3] H. Martin, L. Hermerschmidt, D. Kerpen, R. Häußling, B. Rumpe and K. Wehrle, 'A comprehensive approach to privacy in the cloud-based

Internet of Things', Future Generation Computer Systems, vol. 56, pp. 701–718, 2016.

[4] J. Singh, T. Pasquier, J. Bacon, H. Ko and D. Eyers, 'Twenty security considerations for cloud-supported Internet of Things', IEEE Internet of Things Journal, vol. 3, no. 3, pp. 269–284, 2016.

[5] M. Marwan, A. Kartit and H. Ouahmane, 'A Cloud-based framework to secure medical image processing', Journal of Mobile Multimedia, vol. 14, no. 3, pp. 319–344, 2018.

[6] S. Sicari, A. Rizzardi, L. Grieco and A. Coen-Porisini, 'Security, privacy and trust in Internet of Things: the road ahead', Computer Networks, 2014, vol. 76, pp. 146–164.

[7] J. McLean, 'Security models', Encyclopedia of Software Engineering, Wiley & Sons, 1994.

[8] A. Azaria, A. Ekblaw, T. Vieira and A. Lippman, 'MedRec: using blockchain for medical data access and permission management', In Proceedings of 2nd International Conference on Open and Big Data, pp. 25–30, 2016.

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

[10] A. Zhang and X. Lin, 'Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain', J. Med. Syst. vol. 42, no. 8, 2018.

[11] M. Ed-Daibouni, A. Lebbat, S. Tallal and H. Medromi, 'Toward a new extension of the access control model ABAC for cloud computing', Lecture Notes in Electrical Engineering, Springer, vol. 366, 2016, pp. 79–89.

[12] F. Sifou, A. Kartit and A. Hammouch, 'Different access control mechanisms for data security in cloud computing', In Proceedings of the International Conference on Cloud and Big Data Computing (ICCBDC), 2017.

[13] R. Sandhu, 'Engineering authority and trust in cyberspace: the OM-AM and RBAC way', In Proceedings of the fifth ACM workshop on Role-based access control, Berlin, Germany, 2000, pp. 111–119. Doi:https://dl.acm.org/citation.cfm?id=344309.

[14] A. Ouaddah, A. Abou Elkalam, A. Ait Ouahman, 'Fair access: a new blockchain-based access control framework for the Internet of Things',

Security and Communication Networks, vol. 9, no. 18, pp. 5943–5964, 2017.

[15] P. Samarati and S. Capitani de Vimercati, 'Access control: policies, models, and mechanisms', In Proceeding of the International School on Foundations of Security Analysis and Design, pp. 137–196, 2000.

[16] B. J. Garback and A. C. Weaver, 'XACML for RBAC and CaDABRA: constrained delegation and attributebased role assignment', Computer Science Department, University of Virginia, 2005. Available: https://www.cs.virginia.edu.

[17] A. Bertolino, F. Lonetti and E. Marchetti, 'Systematic XACML request generation for testing purposes', In Proceedings of the IEEE International Conference on Software Engineering and Advanced Applications (SEAA), pp. 3–11, 2010.

[18] N. Rifi, N. Agoulmine, N. Chendeb Taher and E. Rachkidi, 'Blockchain technology: is it a good candidate for securing IoT sensitive medical data?', Wireless Communications and Mobile Computing, vol. 2018, 2018.

[19] E. Yuan and J. Tong, 'Attributed based access control (ABAC) for web services', In Proceedings of the IEEE International Conference on Web Service, 2005. doi:10.1109/ICWS.2005.25.

[20] W. W. Smari, P. Clemente and J. F. Lalande, 'An extended attribute based access control model with trust and privacy: application to a collaborative crisis management system', Future Generation Computer Systems, vol. 31, pp. 147–168, 2014.

[21] I. Sukhodolskiy and S. Zapechnikov, 'A blockchain-based access control system for cloud storage', In Proceedings of IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering, pp. 1575–1578, 2018. Doi:10.1109/EIConRus.2018.8317400.

[22] H. Dukkipati, Y. Zhang and L. C. Cheng, 'Decentralized, blockchain based access control framework for the heterogeneous Internet of Things', In Proceedings of the third ACM Workshop on Attribute-Based Access Control (ABAC'18), pp. 61–69, 2018.

[23] H. ES-Samaali, A. Outchakoucht and J. P. Leroy, 'A block-chain based access control for Big-data', International Journal of Computer Networks and Communications Security, vol. 5, no. 7, pp. 137–147, 2017.

[24] G. Zyskind, O. Nathan and A. Pentland, 'Decentralizing privacy: using blockchain to protect personal data', In Proceedings of IEEE Symposium on Security and Privacy Workshops, San Jose, USA, pp. 180–184, 2015.

[25] K. Yang and X. Jia, 'ABAC: Attribute-based access control', Security for Cloud Storage Systems. SpringerBriefs in Computer Science. Springer, pp. 39–58, 2013.

[26] N. W. Lo, T. C. Yang and M. H. Guo, 'An attribute-role based access control mechanism for multi-tenancy cloud environment', Wireless Personal Communications, vol. 84, no. 3, pp. 2119–2134, 2015.

[27] A. Reyna, C. Martín, J. Chen, E. Soler and M. Díaz, 'On blockchain and its integration with IoT: challenges and opportunities', Future Generation Computer Systems, vol. 88, pp. 173–190, 2018.

[28] A. Outchakoucht, H. ES-Samaali and J. P. Leory, 'Dynamic access control policy based on blockchain and machine learning for the Internet of Things', International Journal of Advanced Computer Science and Applications, vol. 8, no. 7, pp. 417–424, 2017.

[29] H. Kaur, A. Alam, R. Jameel, A. K. Mourya and V. Chang, 'A proposed solution and future direction for blockchain-based heterogeneous medicare data in cloud environment', Journal of Medical Systems, vol. 42, no. 8, article 156, 2018.

[30] Bitcoinj Java Library (accessed July 2020) https://bitcoinj.github.io/.

[31] F. Tschorsch and B. Scheuermann, 'Bitcoin and beyond: a technical survey on decentralized digital currencies', IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2084–2123, 2016.

[32] J. A. Garay, A. Kiayias and N. Leonardos, 'The Bitcoin backbone protocol: analysis and applications', In Proceedings of EUROCRYPT, LNCS, Springer, vol. 9057, pp. 281–310, 2015.

[33] M. Crosby, P. Pattanayak, S. Verma and V. Kalyanaraman, 'Blockchain technology: beyond bitcoin', Applied Innovation, vol. 2, pp. 6–10, 2016.

[34] K. Christidis and M. Devetsikiotis, 'Blockchains and smart contracts for the Internet of Things', IEEE Access, vol. 4, pp. 2292–2303, 2016.

[35] https://bitcoinj.github.io/javadoc/0.14.6/org/bitcoinj/wallet/Wallet.html.

[36] S. Godik and T. Moses, 'Oasis extensible access control markup language (XACML)', ASIS Committee Secication cs-xacml-specication 1.0, 2002.

[37] E. Bertino, S. Castano and E. Ferrari, 'On specifying security policies for web documents with an XML-based language', In Proceedings of

the Sixth ACM Symposium on Access control models and technologies, pp. 57–65, 2001.
[38] Security Policy Tool, available: https://securitypolicytool.com/
[39] M. Drozdowicz, M. Ganzha and M. Paprzycki, 'Semantically enriched data access policies in eHealth', Journal of Medical Systems, vol. 40, no. 11, 2016. Doi:10.1007/s10916-016-0581-7.

Biographies

Mbarek Marwan received an Engineer degree from ENIM, Morocco. He obtained his Ph.D. in Mathematics and Computer Science from ENSA, Chouaïb Doukkali University, Morocco. His area of research covers the latest advances in Cloud Computing, Big Data and IoT.

Abdelkarim Ait Temghart had obtained his Master degree in Business Intelligence from FST-Beni Mellal. He is currently pursuing his doctoral studies in Game Theory, Cloud Computing and IoT at University of Sultan Moulay Slimane, Morocco.

Fatima Sifou earned a Master degree in Software Development from the Faculty of Sciences. He obtained his Ph.D. in Computer Science from University of Mohammed V, Rabat, Morocco. Her research focused mainly on the security and privacy in IoT and Cloud Computing.

Feda AlShahwan received Ph.D. from University of Surrey, UK. She is currently an Assistant Professor at Public Authority for Applied Education and Training, Kuwait. Her current research interests lie in the use of the emerging technologies like Mobile Web Services, IoT, Cloud Computing, etc.