A Smart Contract-based Access Control Architecture for Telemedicine During Covid-19

hideyat zerga (✉ hideyat.zerga@gmail.com)

Université Abou Bekr Belkaid Tlemcen: Universite Abou Bekr Belkaid Tlemcen <https://orcid.org/0000-0002-6451-8269>

Asma AMRAOUI

Abou Bekr Belkaid University Tlemcen: Universite Abou Bekr Belkaid Tlemcen

badr BENMAMMAR

Abou Bekr Belkaid University Tlemcen: Universite Abou Bekr Belkaid Tlemcen

Research Article

Keywords: Internet of Things, Fog computing, Blockchain, Smart contract, Access control, Telemedicine, Covid-19

Posted Date: October 18th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-982468/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

A smart contract-based access control architecture for telemedicine during covid-19

Hideyat ZERGA, Asma AMRAOUI, Badr BENMAMMAR

Telecommunication Laboratory Tlemcen (LTT)

University of Tlemcen (Algeria)

{Hidayet.zerga@ univ-tlemcen.dz , asma.amraoui@univ-tlemcen.dz,
badr.benmammar@ univ-tlemcen.dz }

Abstract. In the fight against the COVID-19 epidemic that is currently a major global public health issue, social distancing has been imposed to prevent the massive transmission, thus doctors in hospitals have turned to telemedicine in order to be able to monitor their patient notably those suffering from chronic diseases. To do so, patients need to share their physiological data with doctors. In order to share this data safely, prevent malicious users from tampering with it and protect the privacy of patients, access control becomes a fundamental requirement.

In order to set up a real-time (Internet of Thing) IoT enabled healthcare system (HS) scenario like telemedicine, Fog computing (FC) seems to be the best solution comparing to Cloud computing since it provides low latency, highly mobile and geo-distributed services and temporary storage.

In this paper, the focus is on access control in the telemedicine systems. Our proposal is based, on one hand, the concept of Fog computing to ensure the distributed aspect needed in the monitoring of patient health remotely; and on the other hand Blockchain (BC) smart contracts, in order to provide a dynamic, optimized and self-adjusted access control.

KEYWORDS: Internet of Things; Fog computing; Blockchain, Smart contract; Access control; Telemedicine; Covid-19;

1 INTRODUCTION

The emergence of the new health crisis has played the role of an accelerator, making it possible to cross in a few days stages that usually mark out the cycle of adoption of an innovation. It was necessary to reduce the exposure of professionals and their patients to the virus. And the call by the authorities to consult only in emergencies was so well heard that general practitioners, and medical specialists in hospitals saw their offices deserted overnight. They are logically worried about the follow-up of their patients, in particular those who suffers from chronic diseases (diabetes, heart failure, etc.), whose risks are increased in the context of Covid-19. Thus, they turned to telemedicine.

Telemedicine enables patients to connect with their healthcare providers at distance, to receive treatment in their homes where their health is continuously monitored, to enable doctors to follow the hospitalization process and make recommendations to patients and their supervisors [1].

The IoT in healthcare has been suggested as a promising way to dramatically improve the efficiency and quality of patient care [2-4]. The introduction of Fog computing in IoT-based HS has brought computing resources and storage closer to the data source. As a result, some data processing can be executed closer to the data source, distributing resource demands, reducing the need for multi-hop data communication, reducing latency and energy consumption, and promoting service flexibility [5-8]. Medical devices in IoT-based HS (notably in telemedicine) measure patient vital signs and aggregate this data into medical records that are uploaded to distributed Fog servers for storage and accessed by healthcare professionals. However, these medical records contain sensitive physiological information, and in order to protect patient privacy, access control to outsourced files is essential to prevent unauthorized access to data.

Traditional access control such as Discretionary Access Control (DAC), Mandatory Access Control (MAC), attribute-based access control (ABAC), role-based access control (RBAC), capability-based access control (CBAC) and usage control based-access control (UCON) are unable to provide an efficient mechanism to meet the requirements of IoT systems, as they use a centralized access control

approach in which the centralized server handles all user access requests. This can cause a single point of failure (SPOF) [9-11]. The access control system should be distributed to solve this issue, flexible and scalable to accommodate a large number of users, lightweight to accommodate the IoT resource constraint, and located at the edge of the network to provide a real-time response [12,13]. In order to meet all these requirements, we will integrate the Blockchain technology to the Fog-IoT healthcare Architecture.

Blockchain is an immutable timestamp ledger of blocks that are used for storing and sharing data in a distributed manner by a peer-to-peer network [14]. Blocks in Blockchain are shared across all participating nodes, which eliminates the need for a central authority, it also improves transaction speed, by removing the delay introduced by the central authority and at the same time, it makes transactions cheaper. A BC platform, when properly designed, can allow cross-domain data sharing and ensure the patients to have secure access to their medical records [15].

Blockchain smart contract is proposed as a solution to overcome the security issues encountered in the IoT access control systems. Smart contracts are functions that can be written into the BC and then executed by all the nodes of the BC, they are computer protocols that facilitate, verify and execute the negotiation or execution of a contract, or that render a clause useless contractual [16].

The integration of BC and FC enable IoT systems applications capitalize on a decentralized structure. On one hand as FC possess a distributed computing environment, it is impeccable to use distributed security mechanisms to secure network resources and data transactions. Therefore, Blockchain technology offers good grounds for Fog-enabled IoT systems to build and manage access control solutions [17]. On the other hand, with the FC, small devices such as smartphones, tablets, and various other smart devices can act as nodes on a Fog computer network. These devices can thus be integrated into the Blockchain and perform access control [18,19].

Our proposed access control is done through several smart contracts. Each resource owner establishes one smart contract that contains all his access policies and then posts it to the Blockchain, to give the patient a full control on his data. Fog servers and nodes will be integrated into the Blockchain but not IoT devices (due to their constrained nature). Fog nodes will be responsible for managing access control permissions for IoT devices while Fog servers will act as minors.

The main contributions of our paper are as follows:

1. To reduce latency and energy consumption in telemedicine we propose a Fog-IoT healthcare architecture
2. To integrate a large number of user devices to our access control, and thus improve it while reducing operating cost, we combined the FC technology with the Blockchain technology.
3. We propose a smart contract-based distributed and dynamic access control mechanism that enables patients to be the real owners of their sensitive data and securely share them.
4. We implement our scheme on a private Ethereum BC. The performance and security analysis clearly prove that our access control is efficient for real-time IoT enabled HS such as telemedicine.

The remainder parts of this paper are organized as follows: Section 2 gives a view on the related work. Section 3 explains our proposed system architecture for a telemedicine scenario in detail. Section 4 introduces our proposed smart control framework. The implementation and performance analysis are given in section 5, and finally, this paper is concluded in Section 6.

2 Related work

A. Blockchain based access control

In [20], authors have distinguished two models of access control based on BC, transaction-based access control and smart contract-based access control. A transaction is a transfer of values between different entities that are signed, broadcast and finally collected in blocks. Whereas the smart contract is an autonomous script stored on the Blockchain with a unique address. A contract can be created by any user and posted as a transaction in the BC to carry different operations such as program execution,

storage, or an account balance. The contract's code is executed whenever it is called from a user or from another contract. Each smart contract interaction is considered as a transaction and will be validated and added to the Blockchain. In [21], authors defined an approach to create, manage and enforce access control policies exploiting transaction-based access control. The access model includes two types of transactions. Policy Creation Transaction, and Right Transfer Transaction. The first one is established by the resource owner to generate and transfer an access token if the subject attributes satisfy the access policies. The second one is used when the access token is passed from the current subject to another one. However, access decision is made by an external authorization system.

A scalable access management in IoT has been proposed in [22], where the authors used a single smart contract deployed by the agent node, that contains all the functionalities of system access management, however, this approach deprive the resource owner from controlling his own data. In our framework we give to the patient a full control of his data. A smart contract-based access control for IoT combined with machine learning algorithms is proposed in [23] in order to provide dynamic access control by verifying the behavior of the subject. The architecture of the system consists of several access control contracts (ACC), a judge contract (JC) and a registry contract (RC). Each ACC defines an access control method for a subject-resource pair. For each resource, a list of bad behavior is stored in the ACC contract. When called by a subject to gain access, ACC is executed and if a bad behavior is detected, ACC reports it to the JC. Based on a misconduct judgment method, the JC determines the corresponding penalty as blocking access to the subject for a period or allowing access if the ACC contract is correctly executed and has not detected any misconduct. However the overhead communication between contacts can increase latency from our point of view. Smart contract-based access control has also been adopted in [24][25]. In [25] contract fulfillment generates feedback on the current state of the reinforcement learning (RL) based system and send it to the resource owner to improve or upgrade the security policy dynamically.

In HS, some initial work has also been done on the BC based-access control. In [26], the authors proposed a smart contract-based access control framework where they used four forms of smart contracts for user verification, access authorization, misbehavior detection, and access revocation respectively. Considering the block size of ledger and huge amount of patient data, the EMRs are stored in Cloud after being encrypted, while their corresponding hashes are packed into BC. However, the use of Cloud computing for storage retrieve of data from the Cloud increases latency, in our work we are using FC in our architecture to reduce the latency needed in real-time applications such as telemedicine. A Blockchain-based architecture for e-health applications is proposed in [27], where they modify the classic BC structure in order to overcome its challenges in IoT applications. To this end, they cluster the miners of BC, store and process data at the nearest cluster to the patient. Another BC framework for managing electronic health records named Ancileis proposed in [28].

B. Fog computing architecture based Blockchain

Some initial work has been done on the integration of Blockchain-Fog computing architecture, In [29], authors designed a FC model based on the BC for Industrial IoT as to make fast and reliable exchange of data based on low throughput and low latency. In [30], the proposed system deploys FC with BC and SDN to support IoT networks and applications, where they used SDN controllers to control and manage Fog nodes, and IoT devices, that are connected by the BC to provide high security level to the IoT network.

In healthcare systems, some initial work has also been done on the integration of Blockchain-Fog computing architecture, authors in [31] proposed an efficient BC-based secure healthcare services for diabetes and cardio diseases prediction in FC, where the patient health information is collected from Fog Nodes and stored on a BC. Fog computing-based Blockchain framework for activity recognition as an application to e-Healthcare services is proposed in [32], their objective is to recognize human activities in any environment that may be real-time and online/offline videos supporting Cloud or Fog

architecture. Integration of FC with BC is also proposed in [33] to overcome the issue of healthcare IoT device identification, authentication, and verification for scalable frequent data transmission in a decentralized environment, where they used an FC-based three-tier architecture.

To the best of our knowledge no one has combined the two technologies to make access control and especially in HS. To address the limitations of the above works. This paper proposes a smart contract-based access control for Fog HS which consists of multiple access control contracts (ACCs) to achieve distributed and trustworthy access control for IoT systems. In our framework, each ACC provides all the access control methods of the subjects that can interact with the object and implements both static access right validation based on predefined access control policies and dynamic access right validation by checking the behavior of the subject.

3 Proposed system architecture

As illustrated in Fig. 1, the architecture of the system considered in our proposed diagram consists mainly of entities such as patients, hospital staff, Fog nodes which are represented by the devices of patients and hospital staff, the numerous IoT devices (sensors and actuators), FC servers, and Cloud server. The main roles of peers are explained below:

- **Patients:** Patients in their homes are an essential part of our system. The access control mechanism is designed to make them the real owners of their data. They don't only have access to their medical records, but they can also manage attempts to access their data through smart contracts.
- **Hospital staff:** The hospital staff in our system includes doctors, nurses, specialists and IT support, who access and share patient data with each other.
- **Medical Sensors and Actuators:** The IoT devices in the system mainly include sensors, which can perceive physiological data (e.g. temperature, heart rate, insulin levels) and send it to the Fog for later use, and actuators, that can perform certain operations (for example, adjust the insulin level in the blood) after they receive a command from users.
- **Fog Nodes:** A Fog node is a user device (desktop computer, laptops, smartphones, and tablet) through which users (patient or hospital staff) can take advantage of services (ex: check current body temperature, check patient's insulin level). These devices support multiple communication protocols to aggregate data from various heterogeneous IoT devices.
- **Fog Servers:** These are geographically distributed nodes. The Fog Server is a lightweight Cloud server that is responsible for collecting, storing, processing and analyzing information from IoT devices and providing predefined applications and services on demand. In addition, he must decide what information should be sent to the Cloud, in what data format and when. Fog resources are integrated with access points, routers and network gateways, alongside generic network functions.
- **Cloud:** The Cloud Data Center is the first platform for IoT-enabled healthcare solutions. In addition to large-scale computing, it facilitates storage, utility services with reliability and scalability. Cloud resources (computing infrastructure, storage, etc.) are structurally orchestrated and are virtualized.

Fig 1. Proposed system architecture

In typical IoT applications, each entity may have some resources (e.g., services, data, storage space) that are needed by the other entities. Thus, access control must be implemented by all resource owners to prevent unauthorized use of their resources especially in HS. Therefore, the access control method must be implemented by the data owners to prevent illegal and unauthorized use of their private medical records. The data owner must be able to recognize the adversary and restrict those requests for illegal access. The requesters have access to the data only after satisfying smart contract-based access control policies explained in detail in Section IV.

4 Proposed smart contract-based access control

This section presents the proposed smart contract-based distributed and patient centric access control framework.

The basic elements of an access control system are the subject, the object and the owner of the object. The first item that is an active item is the subject requesting access to the object. The object represents a passive entity (resource) in a system to which access is requested. Access to an object means access to its information such as records, fields, pages, and programs. Each object has an owner who defines the access policy and grants permissions to the subject. The subject, object and object owner considered in our system are defined in Table 1.

Subject	Object	Object owner
Patient/hospital staff	IoT devices (sensors and actuators)	Patient
Patient/hospital staff	Fog nodes (user's devices)	Patient/hospital staff
Patient/hospital staff	Fog servers	Hospital staff (administrator)
Patient/hospital staff	Cloud server	Hospital staff (administrator)

Table 1. The basic elements of access control considered in our system

Our access control is done through several access control contracts (ACCs). Each resource owner establishes one smart contract that contains his access policies and then publishes it on the BC.

In our proposed system we assume that the entity registration process is already done. The patient, hospital staff, IoT devices, nodes and Fog servers are registered in the hospital identification system and have a unique ID number, Pid, Hid and Rid are the unique identifiers of the patient, hospital staff and resource respectively. After completing the registration process, each entity has a respective unique address in the BC corresponding to its unique ID.

Our Blockchain-based access control goes through the following steps:

- **Step 1** (creation of smart contracts): the owner of the resource creates a smart contract and sends a transaction to deploy it on the BC.
- **Step 2** (Submit the request): the subject submits a request to access an object. In case of an IoT device, the request is forwarded to the associated Fog node.
- **Step 3** (contract retrieval): The Fog node requests to retrieve the according access control contract.
- **Step 4** (Transactions are recorded in the new block): The Fog node sends a transaction which contains the information necessary for access control. The transaction will be mined and encapsulated in a new block, then the access control methods can be executed.
- **Step 5** (Longest new legal string): As soon as the access control process is finished, the result is sent to the subject and the object.

In the first step, when creating the smart contract, the owner of the object will have to define the access policies to his resource by filling in the subject address, the type of resource which the subject can or can't access such as a record, a field or a program, the action that the subject can perform on it object like reading the data or modifying it or executing a program, and finally the access permission which can be either allow or deny. The Table2 is an example of a list of access policies filled out by an object owner.

Subject address	Resource	Action	Permission
0x5fF07...	File A	write	allow
0x5fF07...	File A	read	allow
0x9f803...	Program B	execute	deny
0xFf917...	Folder C	write	allow

Table 2. Access policy list of an object

When the subject submits a request to access an IoT device, the request is forwarded to the associated Fog node that will retrieve the according ACC (we assume that the Fog knows its address). The access control contract receives a transaction from the subject which contains the following information: subject address, the resource to which he wants to access, the action he wants to perform and the time of access request ToRS, then the access control methods can be executed.

It should be noted that in our system, ACCs also provide functions for adding, updating and deleting access control policies.

- **policyAdd ()**: This ABI(Application Binary Interface) receives the information set by the object owner and adds them to the list of policies.
- **policyUpdate ()**: This ABI receives information set by the object owner to update the policy.
- **policyDelete ()**: This ABI is invoked by the object owner to delete a policy.
- **accessControl()**: This ABI receives the information required for access control and returns the result of the access and the penalty. When the subject calls (by sending a transaction) this ABI to authorize their current access request, the validation process begins. In case of a validation failure or once a possible bad behavior is detected, the ACC makes a penalty decision based on the misconduct history and applies countermeasures based on the penalty decision.

- **deleteACC ():** This ABI is invoked by the object owner (creator of the ACC), it performs the self-destruct operation to remove the ACC code and storage from the BC, so that the ACC can no longer be available.

Our access control goes through three stages, validation of access request, policy check, and misconduct check.

Validation of Access request: During this process, the ACC verifies the ID of the subject, if it is validated, the contract goes to the policy check otherwise the request is rejected.

Policy Check: The contract checks in the list of access policies whether the subject has the right to access the resource or not, if so, he passes to the misconduct check otherwise the request is rejected.

Misconduct Check: The access control contract maintains a list of misconduct for each subject as shown in Table 3. If the subject has made more frequent access requests than they are authorized, the request is rejected otherwise the request is approved and the subject can access the object.

Resource	Action	Time of misconduct	Penalty	Time of Unblock
Folder A	read	01/02/2021; 11:45	1min	01/02/2021; 11:46
Program B	write	01/02/2021; 12:30	2min	01/02/2021; 12:32

Table 3. List of misconduct for subject A

5 IMPLEMENTATION AND PERFORMANCE ANALYSIS

This section explains the implementation setup and analysis of our proposed smart contract-based access control scheme. It also describes the hardware and software used to perform the experiments and finally, a performance analysis is made.

A. Implementation

In our framework, we treat every device as an account and every access control request as a transaction. Servers and Fog Nodes are nodes belonging to the BC, the servers act as miners, and each Fog node is considered a lightweight BC node, it does not participate in the mining process, it is responsible for managing access control permissions for IoT devices. IoT devices are not part of the BC due to their restricted nature and we assume that they are trusted entities. Fog nodes are used to provide scalability to the system by preventing IoT devices from performing heavy calculations involving a task related to access control and communication with the BC.

1. **ACC:** In the implementation of the ACC, to help characterize the misconduct, we have added the following fields to the rows (i.e., policies) in TABLE II:

MinInt: the minimum time allowed between two successive requests.

NoFR: Number of frequent requests.

Limit: Limit of frequent requests.

We introduced a variable named "timeOfUnblock" for each resource, which is set to 0 when requests are unblocked. We used a struct to store the fields of subject's registration and applied a mapping from the fields of subject address to this struct to construct the subject's registration list. We used another struct to store the fields of a policy and applied a three-dimensional mapping from the fields of subject address (primary key), resource (secondary key) and action (tertiary key) to this struct to construct the subject policy list. And finally, we used a dynamic array to store the misbehavior records of a subject. Based on the above fields and variables, we designed the accessControl ABI as in Algorithm 1, which receives the inputs of subject, resource, action and time and returns the access

result and penalty. When a misconduct detected, we use the following function to determine the corresponding penalty:

$$\text{Penalty} = \text{base} * \text{interval}$$

Where “interval” is the number of misconducts exhibited by the subject and “base” is initialized during the deployment of ACC.

The registration checks is in line 4, the static validation is in Line 7 and the dynamic validation is from Line 9 to Line 21. The event returnAccessResult (result, penalty) in Line 22 is used to return the access result and penalty to both the subjects and the objects.

Algorithm 1 Access control ABI

Input: subject, ressource, action, time
Output: result, penalty
Require: $\text{penalty} \leftarrow 0, \text{result} \leftarrow \text{false}$, subjectpolicyList Policies, subjectRegistrationList Registration, SubjectMisconductList List

```

1:  $R \leftarrow \text{Registration}[\text{subject}]$ 
2:  $P \leftarrow \text{Policies}[\text{subject}][\text{ressource}][\text{action}]$ 
3:  $M \leftarrow \text{List}[\text{subject}]$ 
4: if  $R == \text{true}$  then
5:   if  $M.\text{timeOfUnblocktime}$  then
6:      $M.\text{timeOfUnblock} \leftarrow 0$ 
7:     if  $P.\text{permission} == \text{'allow'}$  then
8:        $F \leftarrow \text{time} - P.\text{ToLR}$ 
9:       if  $F \leq P.\text{MinInt}$  then
10:         $P.\text{NoFR} \leftarrow P.\text{NoFR} ++$ 
11:        if  $P.\text{NoFR} \geq P.\text{Limit}$  then
12:          Detect a Misconduct MSC
13:           $S \leftarrow M.\text{length} + 1$ 
14:           $\text{penalty} \leftarrow \text{base} * S$ 
15:           $M.\text{timeofUnblock} \leftarrow \text{time} + \text{penalty}$ 
16:          push MSC into the SubjectMisconductList
17:        else
18:           $P.\text{NoFR} \leftarrow 0$ 
19:           $\text{result} \leftarrow \text{false}$ 
20: else
21:    $P.\text{ToLR} \leftarrow \text{time}$ 
22: Trigger event ReturnAccessResult(result, panalty)

```

2. **JavaScripts at the subject and object:** To implement our access control framework, we need to create two JavaScript (one at the subject side and another one at the object owner side) using web3.js to interact with the contracts. Through HTTPS, using JSON_RPC, we communicate with the geth node over JavaScript run-time environment in script mode. The access control in this study is implemented based on the case where the ACC is called by the subject and the result is returned to both sides. The JavaScript at the subject side runs the accessControl ABI of the ACC and watches the event returnAccessResult () returned from the accessControl ABI to retrieve the access result.

B. Hardware and software specifications

We simulate our model using a laptop to create multiple virtual Ethereum nodes. Table 4 shows the specifications of the device we have used to perform experiments.

To create an Ethereum node, a geth client is installed on the device, and several nodes are created. With geth client, we configure Ethereum accounts for each node and form a private Ethereum BC. The laptop acts as miner and deploy the contracts by sending transactions to the BC network. Remix [34] has been used to compile the smart contract code and Ethereum wallet has been used to deploy the

contracts. We also use web3.js [35], an Ethereum JavaScript API at the subject side to communicate with the corresponding geth client through HTTPS connections for deploying the contracts and maintaining their various events. On the receiving side, web3.js communicates with the geth client for receiving the access messages and results from the smart contracts.

Smart contracts are executed at every participating node. Each node updates the BC data every time a new block is appended to the BC network. Since each node has the same copy of data, we ensure that the contracts are executed correctly.

Device Model	Processor	Operating system	Installed memory
Dell Latitude E5440	Intel® Core™ I7-4600 CPU @ 2.10GHz 2.70GHz	Windows 7 professional 64 bits	6 Go

Table 4. Device specifications

C. Results and performance analysis

Ethereum BC platform denotes the amount of work done in the form of a unit called gas. In our case study, the required gas to deploy the ACC is 2879606. Notice that we have used the test Ethereum network, these gas values are just test values, not a real cryptocurrency. For a real system, these values can be reduced further by using low cost consensus such as PoS or DPoS.

Based on the code, the hardware and software, we conducted experiments to show the feasibility of the framework for access control. We added a policy to the ACC with "MinInt" set to 100 seconds, a "Limit" set to 2, and a "base" set to 30.

The results in Fig. 2 represent the access denial in the case of validation failure. In this case, the message of "Validation failure! Invalid Requester" is displayed and the access results are shown as "false".

```

PS C:\Users\DELL\private-blockchain1> node subject.js
Send access request?(y/n)y

Contract: 0xe9758782fcec8a126015a26744f2dce6dba8adc2
Block Number: 4895
Tx Hash: 0xea5e21b2060c7f771c02b2c91d8266be23e741d42a0b0805cb05a9de496bf782
Block Hash: 0x58743f697edf702616dc7fb02ce9798cbe80b305be136190bd9e9550590d9c1c
Time: 1615285012
Error Message: Validation failure! Invalid Requester
Result: false

Send access request?(y/n)

```

Fig 2. Validation failure.

The two first access results in Figure 3 shows the outcomes obtained while deploying our framework. Since the subject is allowed to access the EMR file, the message "Access authorized!" appeared.

Furthermore, in the third attempt, the NoFR exceeded the Limit, so a misconduct was detected, the message " Misbehavior detected ! Too frequent access" appeared and the request was blocked for a period of 30 seconds since it's the first misconduct (the Misbehavior list is empty).However, if the

Misbehavior list was not empty, the penalty would have been bigger, depending on the Misbehavior list size.

The last request shows the rejection by the object owner when the "timeOfUnblock" is not reached yet, in this case before the 30 seconds penalty ends, and the message "Request are still blocked" appeared.

```

Contract: 0xaa3d420f5138414840dc2adf0555ad5fe845ed08
Block Number: 9014
Tx Hash: 0xdbfac45e8aef241ccc0cfa6f89f76ae849f1f59bf99563946e9faed578d827d4
Block Hash: 0xcb7a079638bcad3213eaf89181e5a312604d6648c4d99e14ea5bdf9759aa271
Time: 1618750585
Error Message: Access authorized!
Result: true

Send access request?(y/n)y

Contract: 0xaa3d420f5138414840dc2adf0555ad5fe845ed08
Block Number: 9016
Tx Hash: 0xc76177aedc3d7876f583563377eeade649bad89eda6413b9328a42ca64f2102e
Block Hash: 0x2f7e74efe6817bd97f479ee547a38c977b9b145d784c7364abd273a7821bd214
Time: 1618750607
Error Message: Access authorized!
Result: true

Send access request?(y/n)y

Contract: 0xaa3d420f5138414840dc2adf0555ad5fe845ed08
Block Number: 9017
Tx Hash: 0x93f6609a86bfa043967d95ca13d0fe647f713c3cf25b0b92a58fea1339014cc4
Block Hash: 0xecc4f71c22583d31b9812e8418289293c17be04e0e265f9078cc0878d2442da9
Time: 1618750614
Error Message: Misbehavior detected! too frequent access
Result: false
Requests are blocked for 30 seconds!

Send access request?(y/n)y

Contract: 0xaa3d420f5138414840dc2adf0555ad5fe845ed08
Block Number: 9020
Tx Hash: 0x38193b0085c3c200214000b45d6000f570dd6591c05a97dd9c34674c316f479d
Block Hash: 0x701cc3fe567acd02d30d164ba82020504619ba1deb49826b2a05d776f52c53a7
Time: 1618750628
Error Message: Requests are still blocked!
Result: false

```

Fig 3. Access granted and rejected by the subject.

The transaction cost associated with the execution of various functions within the ACC is depicted in Figure 4, while Figure 5 shows the transaction cost of responses received by the subject.

Fig 4. Transaction cost of execution of the ACC functions

Fig 5. Transaction cost of different responses of the ACC.

Compared to the work done in [23,26], we are using less Gas consumption while deploying the smart contracts since we use only one smart contract to do both static and dynamic access control, and only one smart contract by an object owner for all the subjects in the system, additionally, in Fig. 6(a), Fig. 6(b), Fig. 6(c) and Fig. 6(d), we can notice that the transaction cost of the different response of our proposed solution is less than the results achieved by [26]. It is clear that the proposed scheme has the smallest average gas consumption hence, the smallest average access latency or in other words the fastest access-response time.

Fig 6. Comparison between our proposed solution and related work.

6 Conclusion

When the Covid-19 appeared, It was necessary for doctors to turn to telemedicine to limit the transmission of the epidemic. However, this scenario promote the risk of disclosure of patient data. This paper investigated the access control issue in telemedicine during covid-19 epidemic, for which we proposed a smart contract-based framework to implement distributed and trustworthy access control. The framework includes multiple access control contracts (ACCs), one for each object owner that includes all the subjects in the system. The ACC goes through three stages, validation of access request, policy check, and misconduct check. If the two first stages are validated, the ACC judges the misbehavior of the subjects based on the subject's access history during the third stage, and if a misconduct is detected the subject is blocked otherwise he can have access. A case study was also

provided for the access control. The case study demonstrated the feasibility of the proposed framework in achieving distributed and trustworthy access control for the Fog-IoT healthcare systems.

Declarations

Funding: No funding was received for conducting this study.

Financial/non-financial interest: the authors have no relevant financial or non-financial interests to disclose.

Authors contribution: this article is based primarily on H. Zerga thesis.

REFERENCES

- [1] Journal article Moazzami, B., Razavi, K. N., Dooghaie M. A., Farokhi, E., and Rezaei, N. (2020). COVID-19 and telemedicine: Immediate action required for maintaining healthcare providers well-being. *Journal of clinical virology : the official publication of the Pan American Society for Clinical Virology*, 126, 104345. <https://doi.org/10.1016/j.jcv.2020.104345>
- [2] Journal article Gelogo, Y. E., Hwang, H. J., and Kim, H. (2015). Internet of Things Framework for u-healthcare System. *International Journal of Smart Home*, vol. 9, no. 11, pp. 323–330. <http://dx.doi.org/10.14257/ijsh.2015.9.11.31>
- [3] Journal article Nandyala, C. S., and Kim, H. (2016). 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. <http://dx.doi.org/10.14257/ijsh.2016.10.2.18>
- [4] Journal article Dang, L. M., Piran, M.J., Han, D., Min, K., and Moon, H. (2019). A Survey on Internet of Things and Cloud Computing for Healthcare. *Electronics*, 8, 768, pp. 1–49. <https://doi.org/10.3390/electronics8070768>
- [5] Article Firdhous, M., Ghazali, O., and Hassan, S. (2014). Fog Computing: Will it be the Future of Cloud Computing. *Third Int. Conf. Informatics Appl*, pp. 8–15, no. October.
- [6] Journal article Kumari, A., Tanwar, S., Tyagi, S., and Kumar, N. (2018). Fog computing for Healthcare 4.0 environment: Opportunities. *Computer science, Comput. Electr. Eng*, vol. 72, pp. 1–13. <https://doi.org/10.1016/j.compeleceng.2018.08.015>
- [7] Journal article Maksimović, M. (2018). Implementation of Fog computing in IoT-based healthcare system. *JITA (Banja Luka)*, pp. 100–107. <https://doi.org/10.7251/>
- [8] Article Kraemer, F. A., Braten, A. E., Tamkittikhun, N., and Palma, D. (2017). Fog Computing in Healthcare — A Review and Discussion. *IEEE Access*, vol. 5, pp. 9206-9222. <https://doi.org/10.1109/ACCESS.2017.2704100>
- [9] Article Yang, Y., Liu, X., and Deng, R. H. (2017). Lightweight Break-glass Access Control System for healthcare IoT. *IEEE Transactions on Industrial Informatics*, 14, (8), 3610-3617. <https://doi.org/10.1109/TII.2017.2751640>
- [10] Journal article Hu, V. C., Kuhn, D. R., Ferraiolo, D. F., and Voas, J. (2015). Attribute-Based Access Control. *Computer*, February, vol. 48, no. 2, pp. 85-88. <https://doi.org/10.1109/MC.2015.33>
- [11] Article Zhang, P., Liu, J. K., Yu, F. R., Sookhak, M., Au, M. H., and Luo, .X. (2018). A Survey on Access Control in Fog Computing. *IEEE Communications Magazine*, vol. 56, no. 2, pp. 144-149. <https://doi.org/10.1109/MCOM.2018.1700333>
- [12] Article Outchakoucht, A., Es-samaali, H., Abou, A., Kalam, E., and Benhadou, S. (2018). Reinforcement Learning and Blockchain to secure the Internet of Things. *ISTE OpenScience*, pp. 1–20. <https://doi.org/10.21494/ISTE.OP.2019.0324>
- [13] Journal article Agbo, C., Mahmoud, Q.H., and Eklund, J.M. (2019). Blockchain Technology in Healthcare: A Systematic Review. *Healthcare*, 7, 56. <https://doi.org/10.3390/healthcare7020056>
- [14] Article Hölbl, M., Kompara, M., Kamišalić, A., Nemec Zlatolas, and L. (2018). A Systematic Review of the Use of Blockchain in Healthcare. *Symmetry*, 10, 470. <https://doi.org/10.3390/sym10100470>

- [15] Journal article Mcghin, T., Choo, K. R., Zhechao, C., and He, D. (2019). Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications*, vol. 135, no. September 2018, pp. 62–75. <https://doi.org/10.1016/j.jnca.2019.02.027>
- [16] Article Mik, E. (2017). Smart contracts: terminology, technical limitations and real-world complexity. *Law, Innov. Technol.*, vol. 9, no. 2, pp. 269–300. <https://doi.org/10.1080/17579961.2017.1378468>
- [17] Article Tariq, N., Asim, M., Al-Obeidat, F., Zubair Farooqi, M., Baker, T., Hammoudeh, M., and Ghafir, I. (2019). The Security of Big Data in Fog-Enabled IoT Applications Including Blockchain: A Survey. *Sensors*, 19. <https://doi.org/10.3390/s19081788>
- [18] Online document Joshi, N. (October 13, 2019). Fog computing and Blockchain go hand in hand.2019. Retrieved January 4, 2021, from <https://www.allerin.com/blog/fog-computing-and-blockchain-go-hand-in-hand>
- [19] Online document ACH, B. (June 10, 2018). Blockchain et Fog Computing : le réseau de l’Internet des objets.2018. Retrieved January 16, 2021, from https://medium.com/@benjamin_ach/blockchain-et-fog-computing-le-r%C3%A9seau-de-linternet-des-objets-1dfc52370d8f
- [20] Article Riabi, I., and Saidane, L. A. (2019). A survey on Blockchain based access control for Internet of Things. *15th Int. Wirel. Commun. Mob. Comput. Conf*, pp. 502–507. <https://doi.org/10.1109/IWCMC.2019.8766453>
- [21] Article Maesa, D.F., Mori, P., and Ricci, L. (2017). Blockchain based access control. *IFIP International Conference on Distributed Applications and Interoperable Systems*. Springer, Cham, p. 206-220. https://doi.org/10.1007/978-3-319-59665-5_15
- [22] Journal article Novo, O. (2018). Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT. *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184-1195. <https://doi.org/10.1109/JIOT.2018.2812239>
- [23] Journal article Zhang, Y., Kasahara, S., Shen, Y., and Jiang, X. (2019). Smart Contract-Based Access Control for the Internet of Things. *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594-1605. <https://doi.org/10.1109/JIOT.2018.2847705>
- [24] Article Alphand, O., Amoretti, M., Claeys, T., Dall ’Asta, S., et al. (2018). IoTChain: A Blockchain Security Architecture for the Internet of Things. *IEEE Wireless Communications and Networking Conference*, Barcelona, Spain. ffhah-01705455. <https://doi.org/10.1109/WCNC.2018.8377385>
- [25] Journal article Outchakoucht, A., Hamza, E.S., and Leroy, J. P. (2017). 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, p. 417-424. <https://doi.org/10.14569/IJACSA.2017.080757>
- [26] Journal article Saini, A., Zhu, Q., Singh, N., Xiang, Y., Gao, L., and Zhang, Y., (2021). A Smart-Contract-Based Access Control Framework for Cloud Smart Healthcare System, *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5914-5925. <https://doi.org/10.1109/JIOT.2020.3032997>
- [27] Article Hossein,v, Esmaeili, K., Dargahi, M. E., Khonsari, T, et al. (2019). A Blockchain-based privacy-preserving healthcare architecture. *IEEE Canadian Conference of Electrical and Computer Engineering*. vol. 180, pp. 31-47. <https://doi.org/10.1016/j.comcom.2021.08.011>
- [28] Article Dagher, G. G., Mohler, J., Milojkovic, M., and Marella, P. B. (2018). Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. *Sustain. Cities Soc.*, vol. 39, pp. 283–297. <https://doi.org/10.1016/j.scs.2018.02.014>
- [29] Article Jang, S.H., Guejong, J., Jeong, J., and Sangmin, B. (2019). Fog Computing Architecture Based Blockchain for Industrial IoT. *ICCS. Lecture Notes in Computer Science*, Springer, Cham, vol 11538. https://doi.org/10.1007/978-3-030-22744-9_46

- [30] Article Muthanna, A., Ateya, A., Khakimov, A., Gudkova, I., Abuarqoub, A., Samouylov, K., and Koucheryavy, A. (2019). Secure and Reliable IoT Networks Using Fog Computing with Software-Defined Networking and Blockchain. *J. Sens. Actuator Netw*, 8, 15. <https://doi.org/10.3390/jsan8010015>
- [31] Article Shynu, P. G., Menon, V. G., Kumar, R. L., Kadry, S., and Nam, Y. (2021). Blockchain-Based Secure Healthcare Application for Diabetic-Cardio Disease Prediction in Fog Computing. *IEEE Access*, vol. 9, pp. 45706-45720. <https://doi.org/10.1109/ACCESS.2021.3065440>
- [32] Journal article Islam, N., Faheem, Y., Ud Din, I., Talha, M., Guizani, M., and Khalil, M. (2019). A Blockchain-based fog computing framework for activity recognition as an application to e-Healthcare services. *Future Generation Computer Systems*, Volume 100, pp. 569-578. <https://doi.org/10.1016/j.future.2019.05.059>
- [33] Article Shukla, S., Thakur, S., Hussain, S., Breslin, J. G., Jameel, S. (2021). Identification and Authentication in Healthcare Internet-of-Things Using Integrated Fog Computing Based Blockchain Model. *Internet of Things*, Volume 15, 100422. <https://doi.org/10.1016/j.iot.2021.100422>
- [34] Remix- ide for smart contract deployment provided by Ethereum. Available online: <https://remix.ethereum.org>
- [35] Web3 javascript api to interact with ethereum nodes. Available online: <https://github.com/ethereum/wiki/wiki/JavaScript-API>